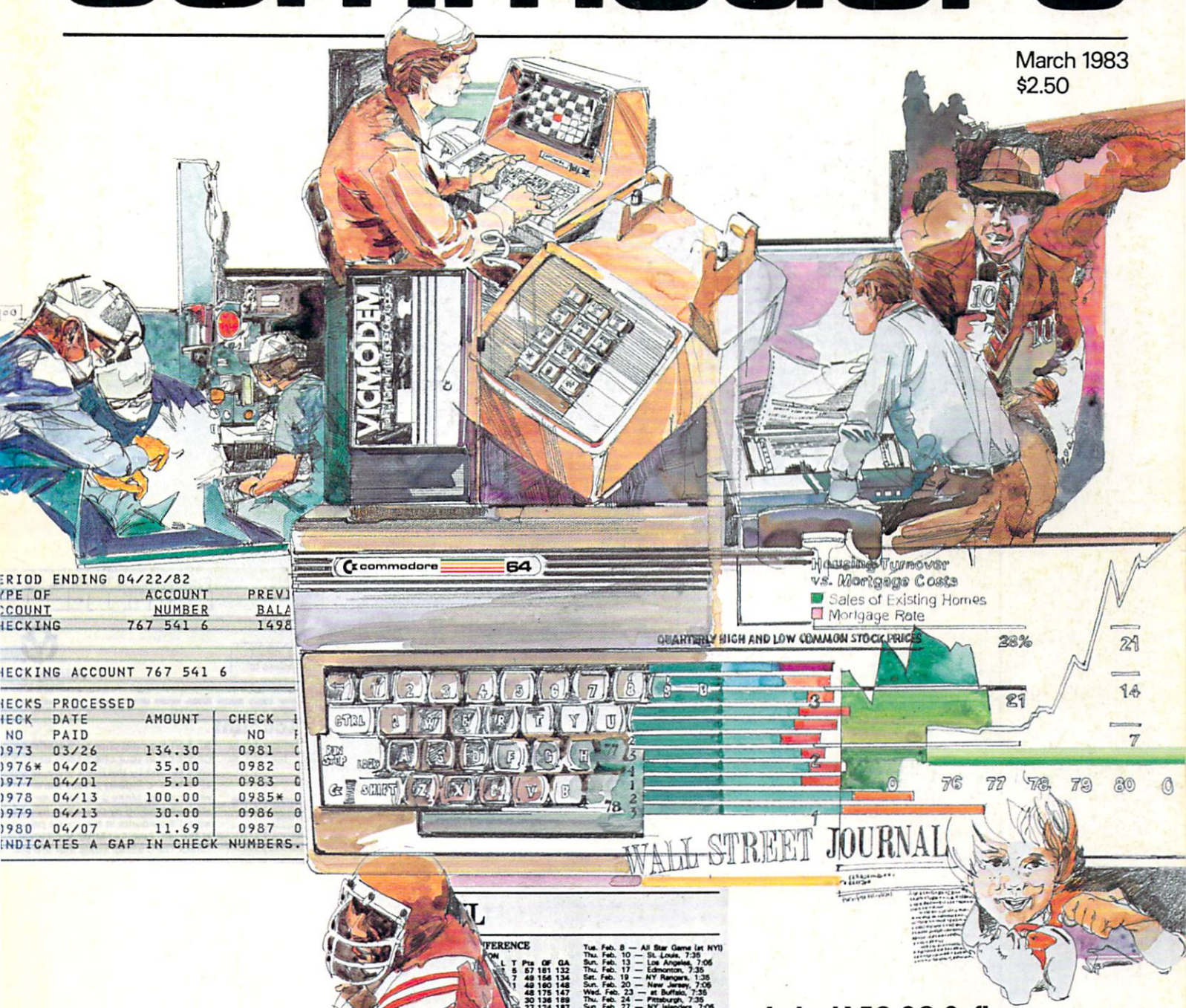
THE MICROCOMPUTER MAGAZINE

commodore

March 1983
$2.50



ERIOD ENDING 04/22/82

YPE OF CCOUNT	ACCOUNT NUMBER	PREVI BALA
HECKING	767 541 6	1498

HECKING ACCOUNT 767 541 6

HECKS PROCESSED

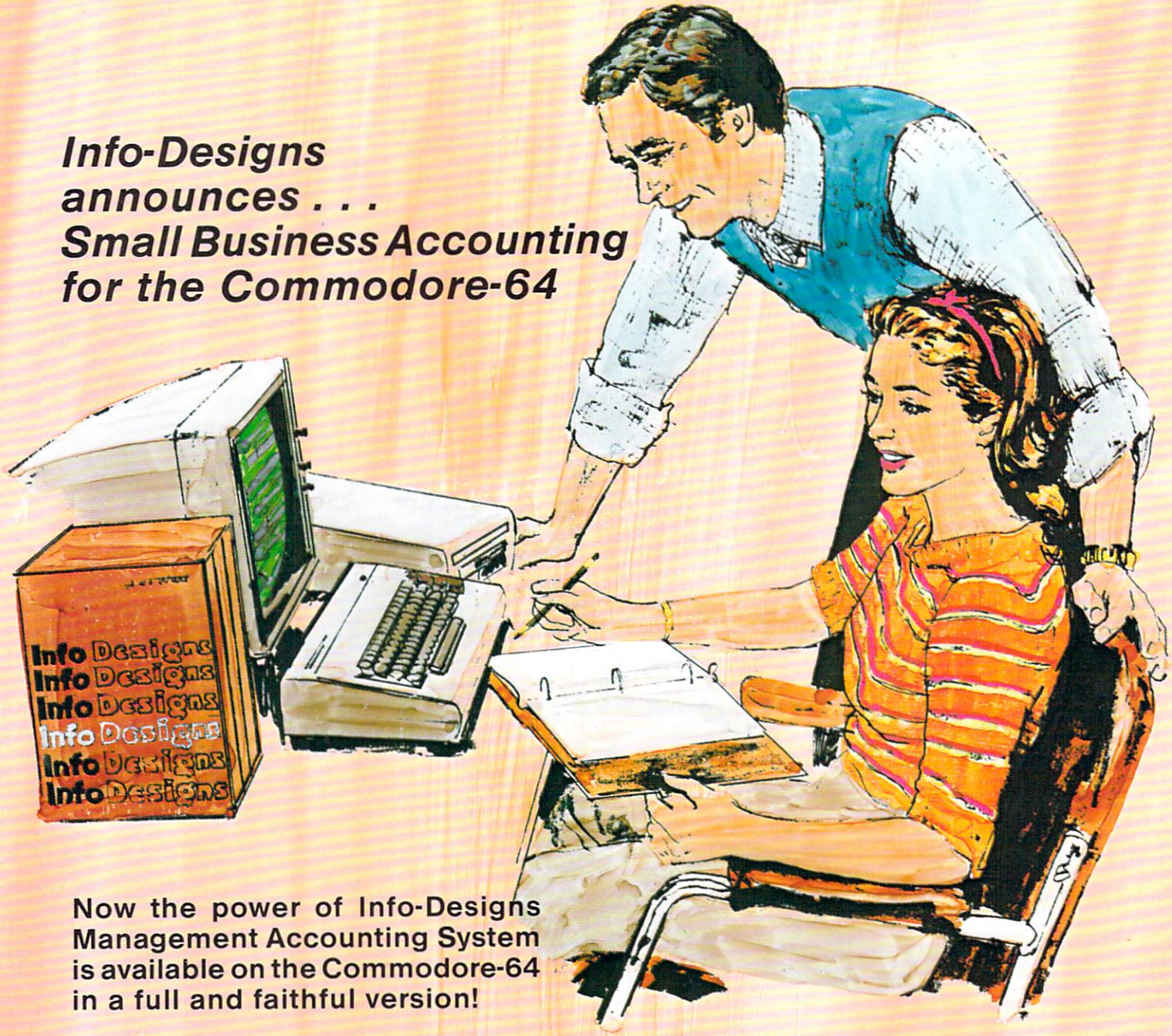
HECK NO	DATE PAID	AMOUNT	CHECK NO	
0973	03/26	134.30	0981	C
0976*	04/02	35.00	0982	C
0977	04/01	5.10	0983	C
0978	04/13	100.00	0985*	C
0979	04/13	30.00	0986	C
0980	04/07	11.69	0987	C

NDICATES A GAP IN CHECK NUMBERS.

- Latest VIC 20 Software
- PET/CBM Disk Quickies
- Commodore 64 as a Synthesizer
- Classroom Networking

Telecomputing

Info-Designs announces . . . Small Business Accounting for the Commodore-64



Now the power of Info-Designs Management Accounting System is available on the Commodore-64 in a full and faithful version!

Thousands of these quality business accounting software packages have been sold on the CBM computer at $595 each. Now, similar features are available to the small business user on the new Commodore-64 for $199 per module!

Select the accounting modules you need—

- Accounts Receivable/Billing
- Accounts Payable/Checkwriting
- General Ledger
- Inventory Management
- Payroll
- Electronic Calendar ($149)

Limited Introductory Offer!

Our SoftPack combination contains the "Big-3" accounting—A/R, A/P and G/L—for only $495. Available for immediate delivery!

Flexible Design

The accounting system will work with one or two VIC-1541 disk drives (or 2031/4040 with IEEE interface), 1525 printer, and color or b&w monitor or TV.

Customer Support Plan

As part of Info-Designs ongoing effort to provide the highest quality microcomputer applications in the marketplace, we offer an optional telephone consulting service to support installation and ongoing operations.

Order NOW. . .for immediate delivery

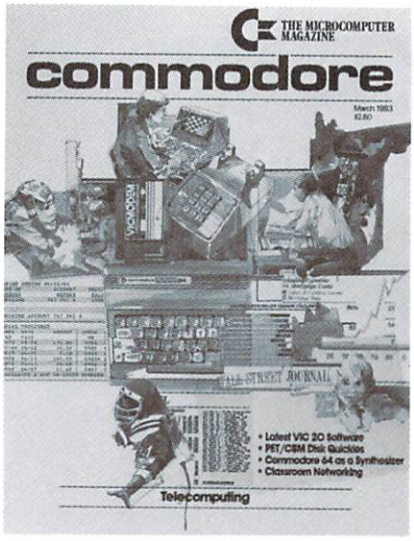
See your local Commodore-64 Dealer or call us directly at **(313) 540-4010**. MasterCard and Visa accepted.

Info Designs

6905 Telegraph Road • Birmingham, MI 48010 • (313) 540-4010

THE MICROCOMPUTER MAGAZINE

commodore



commodore

Publishing Manager
Neil Harris

Editor
Diane LeBold

Staff Writers
Andy Finkel
Paul Goheen
Jeff Hand
Bill Hindorff
Walt Kutz
Ira Neal
Tom Rizol, Jr.
David Rosenwald
Michael Tomczyk

Contributing Writers
Thomas D. Bishop
Elizabeth Deal
Doris Dickenson
Thomas Henry
David Hook
Garry Kiziak
Kent A. Multer
Christopher J. Phillips
Joe Rotello
Lee Silvan
Rick Sterling
John Stockman

Technical Staff
Paul Goheen
Pat McAllister
Cyndie Merten
David Middleton
Tom Rizol, Jr.

Circulation Manager
John O'Brien

Circulation Assistant
Kathy Reigel

Advertising Coordinator
Sharon Steinhofer

Advertising Manager
Diane LeBold

Cover Design
NeumannGreenbergSchlenker

Printing
Oak Lane Printing

Commodore—The Microcomputer Magazine *is published bi-monthly by Commodore Business Machines, Inc., 487 Devon Park Drive, Wayne, PA 19087. Phone: (215) 687-9750. Domestic Subscriptions: six issues, $15.00. Application To Mail At Second-Class Postage Rates Is Pending At Wayne, PA 19087 and at additional mailing offices. POSTMASTER: Send address changes to* **Commodore—The Microcomputer Magazine,** *487 Devon Park Drive, Wayne, PA 19087.*

ADVERTISING POLICY

Advertisements within this magazine are presented for the information of our readers. Acceptance of any ad does not constitute an endorsement by Commodore Business Machines. We stand by only those products manufactured by us. In addition, products displaying the "Commodore Approved" logo have been tested by Commodore.

If any product advertised herein fails to perform as advertised, or you are unable to resolve a problem with an advertiser, please bring the matter to our attention. Write to Commodore Publishing Group, 487 Devon Park Drive, Wayne, Pa. 19087.

COMMODORE PUBLISHING

Commodore is increasingly publishing books relevant to users of our products. We are very interested in hearing from authors and potential authors. Please send a letter outlining your book idea, along with writing samples.

We always need new articles for this magazine. Please send the complete manuscript of the article, printouts of program listings (along with a copy of the program on tape or diskette), and a self-addressed stamped envelope. Include your telephone number in case we have any questions.

King of the mountain!

Workhorse solutions for tough questions.

When **Southern Solutions** acquired the exclusive marketing rights for the CMS Accounting System, the first (and the best) accounting system for the Commodore computer, we offered dealers who were dissatisfied with their current accounting software the opportunity to swap ... ours for anyone else's.

WOW! We were covered with the others ... MAS, BPI, EBS, etc ... all trading for CMS. We provide the only complete coverage of real software for Commodore computers:

THE PREMIER ... SYSTEM IV. Real accounting. More like a mini, yet priced for the Commodore. SuperMath™ gives precision to **$1 billion.** No one else comes close. General ledger, accounts receivable, accounts payable, payroll, inventory, mailing list. Plus important vertical products: oil accounting, pharmacy management, encumbrance accounting, church records and more.

THE STANDARD ... SYSTEM III. Similar to System IV but lower priced. G/L, A/R, A/P, P/R, mailing list.

Commodore 64*. Complete line of bookkeeping record keeping, personal and household management. Usually sells for under $100. Uses one or two drives, just about any printer.

Peripherals. Monitors, monitor cables, blank cassettes.

All software has FileGuard™. Never lose data files, EVEN IF YOU LOSE ELECTRICITY! Compatible with almost any computer, disk drive and printer combination. User-definable reports. Fast file access.

Sold only through professional computer dealers.

To become a **Southern Solutions** dealer, or for the name of your nearest retailer, call or write our General Manager, Bill Swingler.

Dealer Hotline: 1-800-527-4548

*Commodore 64 is a registered trademark of Commodore

Southern Solutions

P.O. Box P, McKinney, Texas 75069 - (214) 542-0278

THE INFORMATION AGE

By Neil Harris

Computers are changing the way we live, a conclusion that comes from observers of the modern world like *Time* magazine and *Fortune*. Both magazines placed the computer at the top of their lists of influences on our lifestyle during the past year.

After all, 1982 was the year that computers appeared everywhere: on the shelves of national retailers and discounters like K-Mart and Montgomery Ward; under the Christmas trees of millions of Americans; and on television commercials and shows.

How are computers changing us? In much the same way as other technological advances have changed us for the past 100 years. The telegraph, telephone, radio, television, and now computers have created an instant worldwide communications system, one in which no part of the world is separate from any other. People the world over are becoming more sophisticated as they gain access to different ideas from perspectives in other places.

We need some way to deal with the incredible mass of information that is building up. The Police, a popular rock band, sing, "Too much information, clogging up my brain. Too much information, making me insane." The problem isn't really in the quantity of information, it's creating any meaning from the bulk of data. Much information is useless, what Arthur C. Clarke calls "noise" in the data gathering systems.

That's where computers come into the picture.

If there is one thing a large mainframe computer is good at, it is taking immense quantities of indigestible data and creating something more usable. Big computers are terrific sorters, shufflers, calculators, and makers of many simple decisions. In short, they are the perfect tool for making sense out of reams of data, to separate good information from garbage.

Since the middle part of this century, computers have become faster, smaller, and cheaper. In their early days, only the government, military, and largest corporations could afford to use computers, which took up whole buildings with special environmental and electrical requirements. Gradually, as the cost of computers decreases, more and more information can be placed into data banks. The result is that information has become more accessible to the average person.

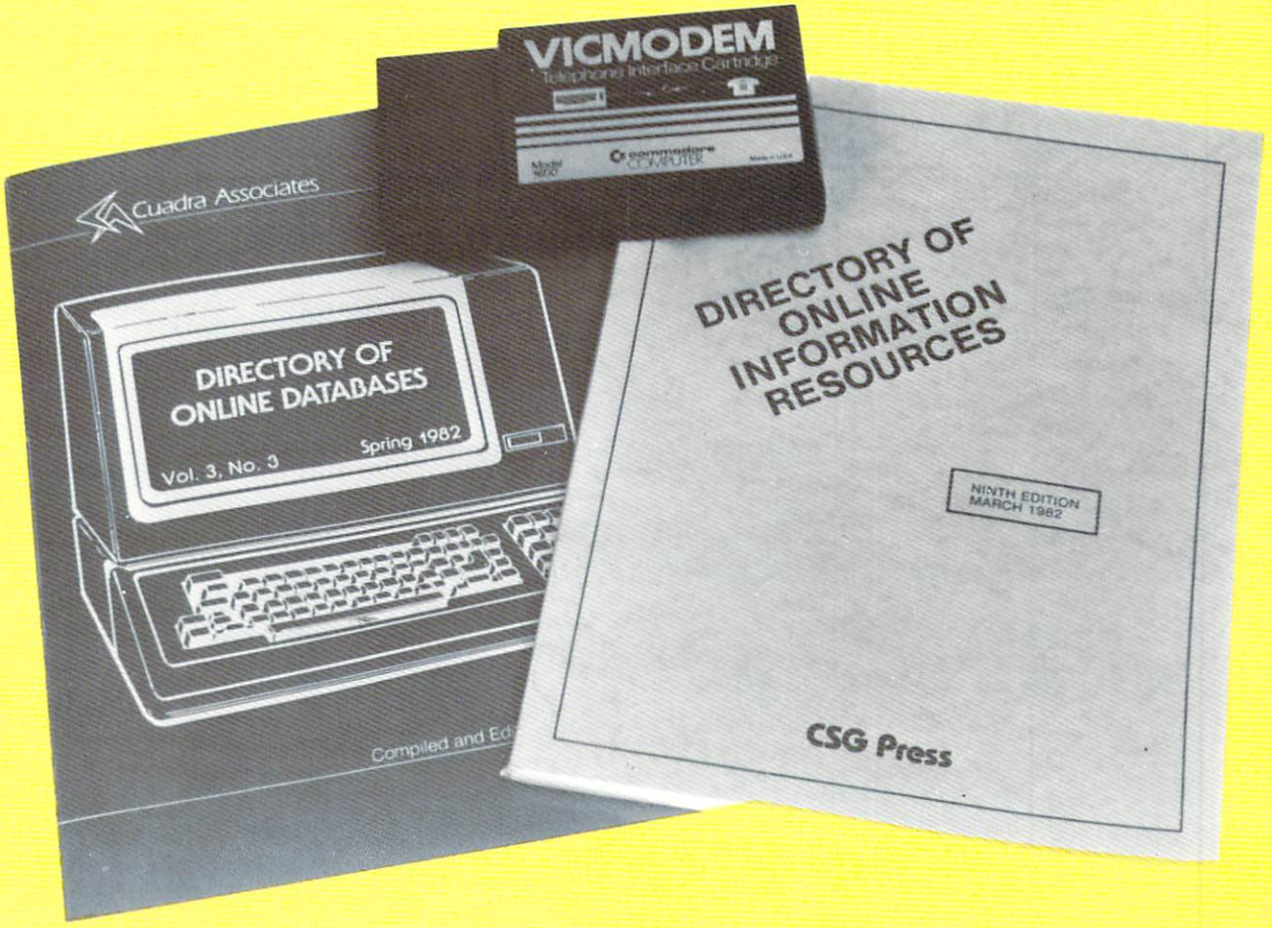
Small businesses now can exercise a much more precise control over their operations than they could ever dream of without computers. They can analyze sales patterns, experiment with different techniques, and determine the best methods for maximizing their profit margins. Computers handle billing, typing, and forecasting. They help the decision-making process.

In the last five years, personal computers have begun carrying this information into the home. Several companies have taken advantage of the growing marketplace of home computerist owners, with information utilities like **CompuServe, THE SOURCE,** and **Dow Jones News/Retrieval** leading the way.

Anyone with access to a telephone can plug their home computer system into these services, using a simple adapter known as a modem (modulator/demodulator). The user pays for the amount of time connected to the system, with time during the day (prime time) costing up to $20 per hour and time at night for under $5 per hour.

At first, the information utilities provided data that was already transmitted through computers. The **UPI** and **AP** newswires and Dow Jones and other stock quotations came first, along with some research services like the **New York Times Consumer Data Bank.** The information utilities found a vast new market in home computer owners for information that was already in their hands.

The types of information available have multiplied in the last couple of years. One recent innovation is the addition of the ***World Book Encyclopedia*** on CompuServe. Anyone on the CompuServe Information System can type the command GO WBE (short for *World Book Encyclopedia*). The computer explains the options available. The user may "feed" the computer a word or phrase, and the appropriate information is found by the computer then reproduced on the screen. Where several choices are possible, the computer gives the user a "menu" from which to choose. The computer responds quickly to inquiries. For the same dollar investment as a printed encyclopedia, a family can buy a computer with a modem and gain access to information that is constantly updated and more varied than any single encyclopedia.



Commodore in involved in this through our **Commodore Information Network** service, also on CompuServe. Any user of Commodore's VICMODEM can access the CompuServe system, type the command GO CBM, and they're in our service. We provide technical information, answers to questions, abstracts from this magazine, and more. See Jeff Hand's article later this issue for more on this.

Specialty companies are zeroing in on specific markets in this business. Games have always been among the most popular pastimes on CompuServe and THE SOURCE, which has led to the emergence of the **GameMaster** system. Located in Chicago, this computer features an innovative operating system that treats the user as if he were a visitor in a house. Each room has a different topic, like war games, card games, meeting rooms, etc. GameMaster users outside the Chicago area must pay long-distance rates to call, but GameMaster's usage rates were recently dropped to $3.00 per hour (any time of the day) to encourage use.

Two specialty information services are **DIALOG** and the **Bibliographic Retrieval Service.** These companies provide information mainly to industrial users and researchers. They are just now starting to respond to the home computer market. They charge very high rates compared to consumer-oriented utilities — anywhere from $10 to several hundred dollars per hour — but the services are very cost-effective compared to the labor of research. Each service holds information from many sources, smaller companies which research only certain topics.

For example, chemists could use the DIALOG system to research all literature indexed by the **Chemical Abstract Service** with over five million references. They can also search through a series of chemical dictionary databases to identify chemical substances from a name segment, synonym, trade name, formula, ring structure, or other fragmentary information.

Our telecomputing department was kind enough to loan me copies of two guides to other information sources: the **Directory of Online Databases** from Cuadra Associates and the **Directory of Online Information Resources** from CSG Press. These guides both provide substantial lists of information resources, with the surprising observation that there is almost no overlap in contents between the two.

Other efforts are under way in other aspects of this information revolution. Project **Xanadu**™ is an effort to make electronic publishing a reality, with special features like hyperstorage™, in which information and cross references stay stable despite changes in storage. Although currently in the development stages, according to Xanadu representative Mark Miller, Xanadu hopes to become "the world's electronic library, the place where mankind stores its literature."

Xanadu was designed as a decentralized system, where the information is duplicated in many different locations. This approach will help to minimize the "library of Alexandria syndrome", where vital information is vulnerable to a single catastrophe. ▶

Xanadu allows users to publish their own materials, paying for their time and storage, which is offset by royalties generated by others reading their material. The material is indexed by the computer into author and title lists, and other indexes can be created by users as profit-generating services.

Even home computers are getting into the information-providing act. Many home systems equipped with auto-answering modems can be set up as bulletin boards and special-interest information exchanges. These services are available without charge (except for the phone call). Among the topics of local services are computer hobbyist information, Bible information, and people's sexual preferences.

The information age isn't coming — it's here! It might not have caught up with you yet, but it won't be long now. The price of information is falling fast. It's up to you to make sure you're equipped to deal with it.

References:

CompuServe, Inc.
5000 Arlington Centre Blvd.
Columbus, OH 43220
(614) 457-8600

THE SOURCE
1616 Anderson Road
McLean, VA 22102
(703) 821-6660

Bibliographic Retrieval Services
1200 Route 7
Latham, NY 12110
(518)783-1161

DIALOG Information Services
3460 Hillview Avenue
Palo Alto, CA 94304
(415) 858-3810

Chemical Abstracts Service
P.O. Box 3012
Columbus, OH 43210
(614) 421-6940 (ext. 3209)

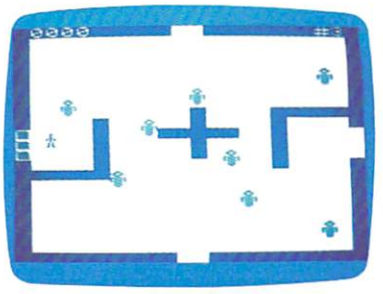
GameMaster
1723 Howard, Room 219
Evanston, IL 60202
(312) 328-9009

Project Xanadu
Box 128
Swarthmore, PA 19081

UMI games...for the fun of VIC™

You've braved the perils of space and landed on the isolated space station AMOK, populated by killer robots! Their only directive is to destroy the invader! The cunning robots stalk you from all sides, evading your most deadly shots. You'll run and shoot from the hip to protect yourself. Stop moving . . . and the diabolical robots will destroy you.

AMOK is just one of UMI's many action-packed arcade-quality games.* This challenging adventure will keep you locked in an hours-long battle for survival against life-threatening robots. United Microware games come on either cassettes or UMI's own durable cartridges, depending on your game selection. If you're ready for thrilling, out-of-this-world adventure, look to UMI. Send for the latest UMI catalog and contact the UMI dealer nearest you today . . . after all, it's just for the fun of VIC™!!



United Microware Industries, Inc.
3503-C Temple Avenue
Pomona, CA 91768 (714) 594-1351

umi

VIC is a trademark
of Commodore Business Machines.

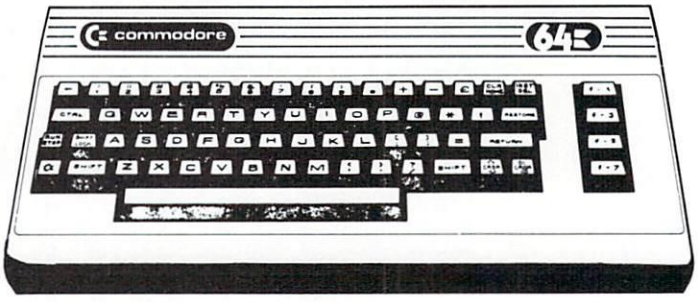
*More action-packed UMI games: Meteor Run — 1613, Renaissance — 1600, Alien Blitz — 1619

BUSICALC

BUSICALC A Honey of an Electronic Spreadsheet

Why electronic spreadsheet programs?

Electronic spreadsheet programs allow the user to create a gridsheet, spreadsheet, worksheet, or any other table of information, using the memory of the computer as pencil and paper. The computer display or terminal acts as a window through which the user views the information as it is entered. Textual information (such as headings), numerical values, and formulas can easily be entered into the spreadsheet.



For Commodore 64™
For Commodore VIC 20™
For Commodore PET®/CBM™ 40 columns
For Commodore CBM 80 column/SuperPet™

BUSICALC Your Computer Drone for Repetitive Calculations

The outstanding advantage of using a computer is that it acts not only as a pencil and paper but as a perfect eraser and an automatic calculator. The user can quickly and easily make any number of alterations to the data within the table. The BUSICALC will evaluate any formula using the data that has been entered. Further, it retains the formulas and displays the resulting value. With BUSICALC controlling the entry of data, providing a comprehensive memory, and performing arithmetic, the preparation of a spreadsheet is faster and more accurate than if it were prepared by hand.

BUSICALC With the Sting Removed from the Prices

BUSICALC 20 . . . . . only **$49.00** for the VIC 20
BUSICALC 64 . . . . . only **$69.00** for the CBM 64
BUSICALC 40 . . . . . only **$79.00** for the original 40 column PET/CBM
BUSICALC 80 . . . . . only **$89.00** for the original 80 column CBMs and SuperPets

BUSICALC AVAILABLE NOW FROM YOUR LOCAL DEALER
(800) 227-9998
FOR THE NAME OF YOUR NEAREST DEALER

California, Canada, Alaska and Hawaii please call (415) 965-1735

Skyles Electric Works
231G South Whisman Road
Mountain View, CA 94041

Europe please contact Supersoft, Winchester House, Canning Road, Harrow Wealdstone, Middlesex, England HA3 7SJ, Tel. 01 861 1166

THE VIDEOTEX Revolution

by Blair Granger and Jeff Hand

To most people in the US today, watching the news on TV means tuning into a nightly newscast, and listening to what the announcer sees fit to tell you is happening. And how many times have you found yourself saying that most of the news you're watching isn't very useful to you? It's not unusual today to watch the news for an hour, and be interested in about ten minutes of it. Well, that situation may soon be a thing of the past for every home in the United States with the advent of the home computer and the videotex phenomenon it has brought with it. And watching the news you want to watch is only a small part of the advantage that videotex provides to make your life alot easier.

The computer itself in one form or another has already been introduced to virtually every American. From the mainframe computer in a large office complex to the Commodore personal computer millions of people use in their homes, to the neighborhood arcade game center, we have all experienced the computer boom. What this boom means now and will continue to mean is mindboggling. Most people think of the computer as something you program, but the things it can do for the average person have nothing to do with whether they know how to program or not.

Picture for a second coming home after work, sitting down at your TV set, giving a few commands to your Commodore computer located next to the TV, and finding on your TV screen a menu of all the latest news from around the world. You simply pick the items you wish to read about and quickly find that news item on your TV. Next you can shop for that new camera you've been wanting, handle your banking, play a game of chess with someone on the other side of the country, and take care of personal and business correspondence on the system wordprocessor. When you're finished, flick off the computer and go on to your evening's plans.

What is it that makes all of this instantaneous information possible? None of the technology is new, but the flexible and inexpensive marriage of personal computers, TV, information science and various modes of transmission make videotex/teletext unique and full of promise. Videotex and teletext can be defined as "a system for the widespread dissemination of textual and graphic information wholly by electronic means, for display on low cost terminals under control procedures easily understood by the untrained user." (Footnote: Tyler, 1979)

To be more specific, videotex is a two-way transmission system, meaning there is interaction between the user and the information source, e.g., CompuServe, THE SOURCE and Dow Jones News/Retrieval, generally broadcast through phone lines or microwaves. Teletext is the transmission of information in only one direction; usually this applies to information broadcast by TV, radio, or microwaves.

Videotex is an idea whose time has come. And the biggest surprise is that it's been happening long before this in other parts of the world, especially cially Europe, Japan, and Canada.

The idea originally started with England's BBC in the mid 70's. They were looking for a way to send cap- ▶

FIGURE 1.2 Videotex via Telephone Line.

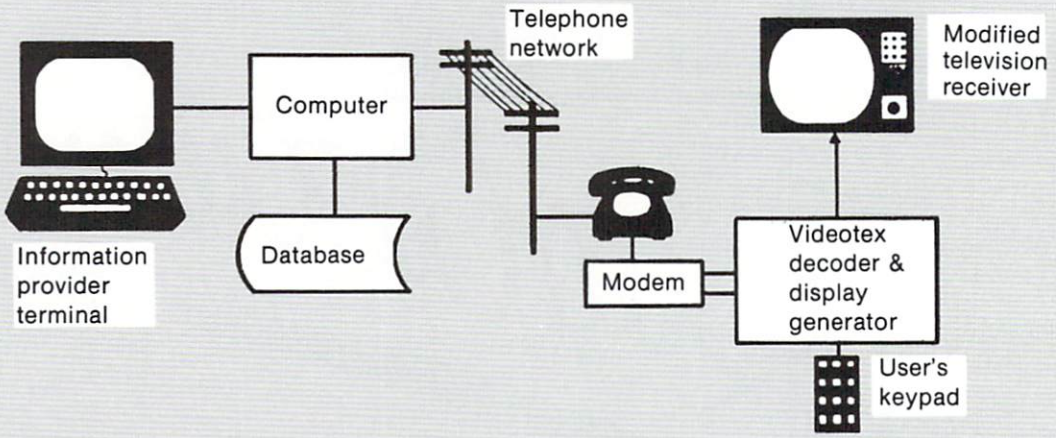
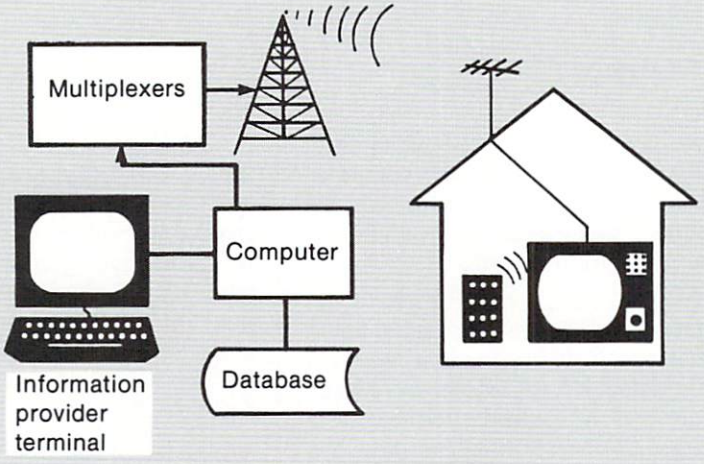


FIGURE 1.1 Broadcast Teletext.



tioned news and subtitles for the hearing-impaired along with the TV transmission. Immediately the network saw the general market potential for this new enhanced method of information transmission, and received backing from the British government to continue development of the technology. The result was CEEFAX (pronounced see facts). CEEFAX provided up to the minute weather, sports and news for everyone.

The Independent TV (ITV) network in England also recognized the value of teletext and soon produced the Oracle. The British post office developed an interactive videotex computer database called Prestel. That system is employed by users to help with tax problems, mortgage calculations, legal advice, and credit card purchases. By 1979 with these three systems in operation the British were leading the way in videotex and teletext, but other nations weren't far behind.

Not to be outdone by the British the French government sponsored development of their own teletext system called Antiope which went into operation in 1977 when the Bourse (French stock exchange) went on the air with stock prices and other financial data. Next the French developed a videotex system called Teletel.

The Canadians decided to outdo both the British and the French with their sophisticated system called Telidon. The Communications Research Centre, the research arm of the Canadian government, developed the system for interactive use over phone lines. The Telidon system is capable of reproducing complicated graphics with high accuracy.

The Japanese, not willing to be outdone by anyone, developed several comprehensive systems but the technology proved to be too expensive for public consumption. Therefore Nippon Telephone and Telegraph settled for CAPTAIN, which has many similarities to Prestel.

Now England, France, and Canada are all selling teletext and videotex wares. England's Prestel was sold to West Germany, Switzerland, Sweden, Finland, Norway, Austria, Netherlands, Belgium and Italy. Videotex/teletext systems are under development in Australia and Hong Kong. And all of these nations are investigating the possibility of an international videotex system for worldwide use.

The question that comes to mind is, why is the U.S. so far behind in videotex development? There are basically two reasons for this. First, videotex development in other nations is intensely supported by the respective governments, but in the

U.S. only private industry has shown any interest in developing videotex. The second reason is that up until the past two years, there was a serious question as to whether a videotex service would sell. Would the consumer need the service? Sure, some businessmen must know the latest stock prices, or whether there is a flight to New York in the next hour, and for them, the high price of a computer was justified. But would the average American spend over a thousand dollars on a computer, just to get the services of a videotex system? Probably not.

However, with Commodore leading the way, computer prices have dropped drastically, so they are easily affordable by everyone. A home computer is very much a practical choice for the home now. Combine this with the desire of Americans to know as much as they can in as short a time as possible, and it is not surprising to see the promise and possibilities of a videotex boom.

If videotex/teletext is a marriage of technologies, in the U.S. we are watching the courtship ritual. Industry wants to see what combination of technologies will produce the best and most profitable system. Therefore test markets and experiments are sprouting up in all levels and types of industries—entertainment, computers, and communications. Some of the companies testing the field presently are: CBS, NBC, Westinghouse, Knight-Ridder Newspapers, Time Inc., Sammons Cable Communications and AT&T. Others such as CompuServe, Dow Jones News/Retrieval Service, and THE SOURCE decided to jump in immediately to get a start on the market.

The size of the impending videotex explosion is forcing many industries to change their business outlook. Many feel that the impact of videotex will be enough to force print publishers to switch from a profit structure based on copies sold to one based on usage. Harper and Row Publishers is making plans that call for a database of various sources from 60 different industries. Many of their reference books will become computerized databases.

Some present trends might be indicators of where the industry is heading. CompuServe is already the largest videotex database for home computers in the U.S. today with over 40,000 subscribers. It also seems fairly obvious that the TV networks will go the route of teletext, having seen the growth of videotex and teletext. Videotex will more than likely zero in on business, educational, and special interest applications. The U.S. has already seen the effects of pay cable television. Given the success of cable it's easy to picture videotex/teletext as a necessary household convenience in a few years.

Unfortunately the main systems like CEEFAX, Prestel, Antiope, and Telidon are not compatible with each other at the present time. No information standard has been agreed upon. And no transmission medium (cable, telephone, TV, and radio transmission) has proven dominant at this time either.

But the heavyweight of the videotex/teletext contenders, AT&T, seems to be putting money and energy into both teletext and videotex technology as well as the development of teletext/videotex standards. AT&T was prohibited from carrying data from mainframes over their telephone lines because of the fear of monopolistic practices. Not until AT&T agreed in late 1980 to sell their rights to local phone companies did the government allow them to carry data over phone lines. Another way that AT&T is testing the new technology is with National Entertainment TV. Ma Bell will use satellite transmission, microprocessors and TV decoders to put the NET signal across the U.S. The teletext signal will be piggybacked on the TV transmission and a jack will be provided in the decoder to hook up a person's home computer.

The AT&T standards mentioned above, called PLP (Presentation Level Protocols), were first introduced at the Videotex '81 convention. The impact of these protocols cannot be underestimated. If there isn't standardization there won't be any extensive videotex system in North America, because if each system requires its unique decoder or terminal the cost of using videotex would be too high to the end user. Many of the competitors in the field, like Knight-Ridder Newspapers, CBS, and Westinghouse, have agreed to make their systems compatible with Bell's PLP. Canada's Telidon system is a subset of PLP and can fairly easily be upgraded to Bell's protocols.

Bell's PLP accommodates ASCII alphanumerics in an eight or seven bit environment. The protocols can be broken down into four functional categories.

1. Display characters that require 40 characters per row and 20 rows per screen.
2. Graphic displays using alphamosaic structures and geometric primitives (lines, circles, etc.)
3. Dynamically redefinable character sets (DRCS).
4. Picture description instructions (PDI).

The mosaics are compatible with those used by the Prestel and Antiope coding schemes. The geometric primitives are compatible with Telidon's picture description instructions.

To get a copy of AT&T's Presentation Level Protocols contact:

Manager Information Planning and Development
AT&T
5 Wood Hollow Road
Parsippany, NJ 07054

We don't recommend getting a copy of the protocols unless you already know something about telecommunications protocols because the reading is dry and definitely written to scare off the beginner.

To see why videotex can be so useful, you have to understand what it can do for you. No one buys computers or technology unless there are benefits, so here are some benefits that might be applicable to you. Most videotex systems provide a wide range of services that cover the following categories: (1) information retrieval, such as news, weather, directories, and bibliographic databases, (2) transactions, like reservations, teleshopping, and telebanking, (3) messaging, for electronic mail, (4) computing, for interactive gaming, programming, and financial analysis, (5) telemonitoring, for home security.

Suppose for example, you're a businessman in a fast moving company. You need to keep on top of current stock quotations while you are in your office or at home. Many might think that you would have to use an expensive service to call up for that information. With the advent of videotex however, there's no problem. Just sign up for a subscription to Dow Jones News/Retrieval at a very reasonable cost and you're set. All you need to do is ask your computer for the latest quotations from the stock exchange and they appear on your screen instantly.

Or suppose you're a housekeeper, and need some new appliances. You could go out to all the different stores and shop around, fighting crowds and traffic. Or with videotex you could sit at your computer terminal at home, type out on your keyboard what you are in the market for, and immediately get a catalogue listing and description of different models of appliances. You can even order them right from your own home. And prices from the catalogue are always well below what you would pay in a store for the same thing. All you need to do all this is a subscription to the shopping service called Comp-U-Store, provided at a a very reasonable fee by the videotex sponsor.

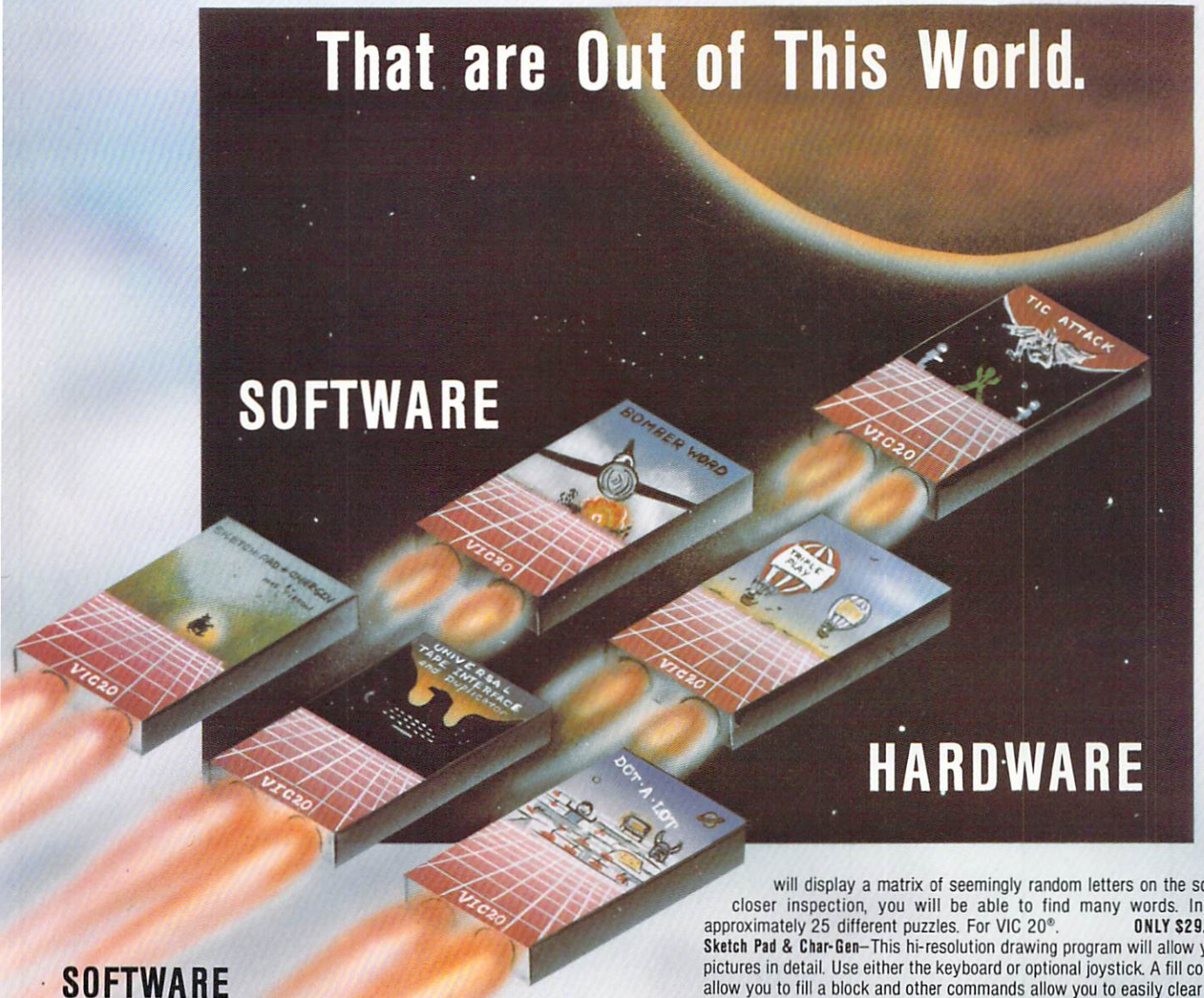
Here is a list of some of the possibilities for videotex:

News
Electronic mail
TV listings
Stock analysis
Gardening hints
Manufacturer's hotline
Best seller list
Bargain finders
Cartoons
Telepoetry
Law
Service manuals
Telebanking
Computer games
Real estate listings
Electronic auctions
Utility metering
Telebetting
Agendas
Bus routes and schedules
File maintenance
Signature samples
Computer art
Word processing
Emergency numbers
Software
Alarm controls
Purchase orders
Reservations
Advertisements
Books
Lobby letters
Legislative meetings
Psychological tests
Teleshopping
Opinion polling
Restaurant menus
Crossword puzzles
Newsletters
Classified ads
Reference materials
Computer aided instruction
Electronic bulletin boards
Medicine
Tax information
Government services
Sports
Library indexes
Purchase orders
Personnel records
Quizzes
Tourist tips
Process control
Children's stories
Music sheets
Evacuation plans
Currency conversion
Community events
Racing forms
Press releases
Inventory control
Soil conditions
Phone listings
Theater guides
Employment opportunities
Opinion
Wine
Groups and clubs

As you can see the list of services available to you is large. With only a basic subscription to a videotex sponsor a large number of those possibilities are yours.

The future looks very promising for videotex. The main ingredients for videotex are now available: Commodore is presently making more home computers than anyone else in the industry and 98% of the homes in the U.S. already have TV (78% are color). It is predicted by industry experts that by 1990 8–10 million homes will have videotex service. This technology has the potential to change the way we use information and eventually have an impact on all of our lives. The future of videotex reaches to the limits of our imagination. And perhaps, there soon won't be any need for a morning newspaper, only a morning videotex briefing!

Products for VIC 20® and Commodore 64®



SOFTWARE

Word Wizard For The Vic 20®–(Requires at least 8K memory expansion) A user friendly WORD PROCESSOR with optional joystick control. Easy edit and string manipulation commands that follow the standard format. Full use of function keys for ease of use. 100% machine language with Delete Word, Search functions and Full Justification. Use VIC Graphic printer, or any centronics compatible printer connected to the user port. On Tape (supports disk). **$34.95.**

Bomber Word–A unique graphic word game on cartridge that provides the full thrill of arcade action. Complete with six modes of play options for added enjoyment. Play against the computer or another player. 6 to adult. For VIC 20®. **$29.95.**

Tic Attack–A fast action arcade game on Cartridge that challenges all of your dexterity. Written in machine language for special audio & visual effects. Over 100 levels of play. High score indication. For VIC 20®. **$29.95**

Dot-A-Lot–As you wander through the maze of life collecting Berries, you happen upon some magical fruit. Pick one and the treasures appear, but the Meanies are out today looking to spoil your fun. Defeat them and continue on to a higher level. An ever changing maze plus arcade type animation and sound will provide a real winning CARTRIDGE for the VIC 20®. **$29.95**

Triple Play–Three word games that are both fun and educational. The games that are included are CROSSWORDS (requires at least 8K expansion). Five complete puzzles are included and each puzzle has up to 100 different words. CRYPTO-SOLVE will help you solve those cryptic messages found in newspapers, books, and magazines with a systematic computer technique. Included are approximately 50 different puzzles. You can even enter your own cryptic messages. HIDDEN WORDS will display a matrix of seemingly random letters on the screen. Upon closer inspection, you will be able to find many words. Included are approximately 25 different puzzles. For VIC 20®. **ONLY $29.95 for all 3**

Sketch Pad & Char-Gen–This hi-resolution drawing program will allow you to draw pictures in detail. Use either the keyboard or optional joystick. A fill command will allow you to fill a block and other commands allow you to easily clear the screen. You can also save and load pictures. Char-Gen is a simple to use custom character generator that will allow you to design different characters for each printable key on the computer. This program is an excellent device to design game creatures, foreign alphabets, secret symbols, or other special characters. One set is included and you can make and store others quite easily. Both for VIC 20®. **ONLY $24.95**

HARDWARE

Expand-O-Ram–16K Expansion Board for the VIC 20® with reset, memory write protect, full memory allocation, plus TWO expansion slots. Like having 2 products in 1 **$149.00**

Universal Tape Interface & Duplicator–(Use on the CBM 64® and VIC 20®). With this device, you can easily load, save or even duplicate tapes easily with your recorder. Full 3 LED indication of Data transfer makes this the most reliable way to Load, Save and Duplicate. A complete I/O device with extras. NOTE: Duplication requires 2 recorders. **Only $49.95**

Universal Parallel Interfaces–Use most any parallel centronics type printer with your Vic 20®/CBM 64®. Two models to choose from to fit your needs. Call or write for information and price.

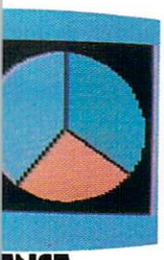
MasterCard VISA

Dealer and Distributor Inquiries Invited

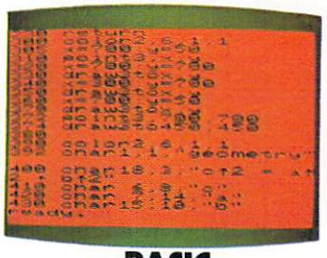
NOTE: We solicit hardware and software items for the VIC 20® and CBM 64® Royalties, license fees, or outright purchases can be negotiated. CBM 64® & VIC 20® are Registered Trademarks of Commodore Business Machines Inc.

MICRO WARE
DISTRIBUTING INC.
1342 B Rt. 23, Butler, NJ 07405
201-838-9027

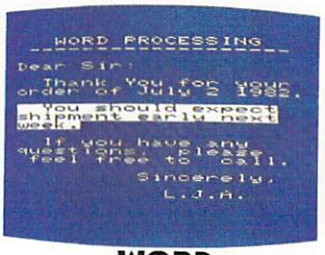
IF YOU OWN A COMMODORE VIC YOU KNOW IT CAN DO ALL THIS.



ENSE
NDAR



BASIC PROGRAMMING

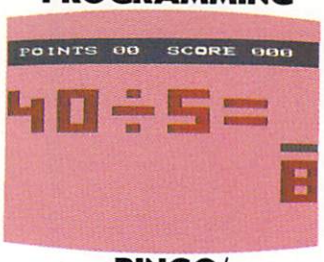


WORD PROCESSING

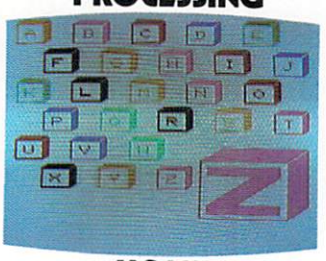
PERSONAL FINANCE

VISIBLE SOLAR SYSTEM

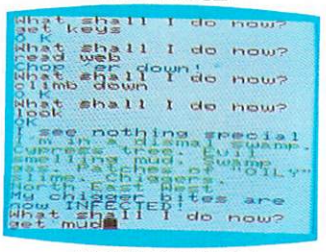
PING
TOR



BINGO/ SPEED MATH

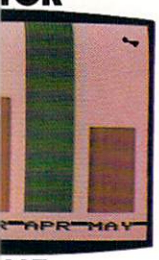


HOME BABY SITTER

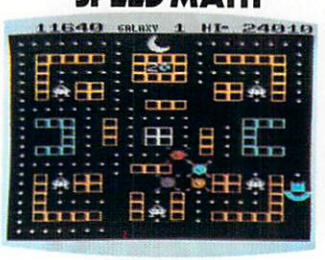


ADVENTURE LAND

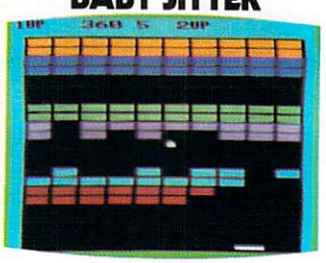
BIORHYTHMS



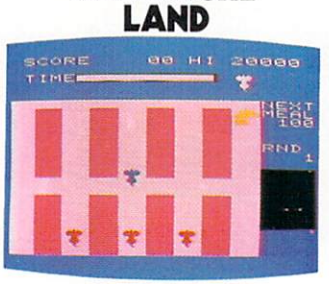
ME
ITORY



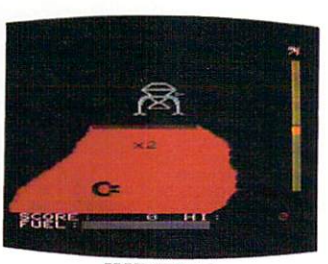
COSMIC CRUNCHER



SUPER SMASH

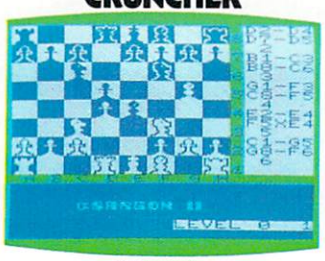


RADAR RAT RACE

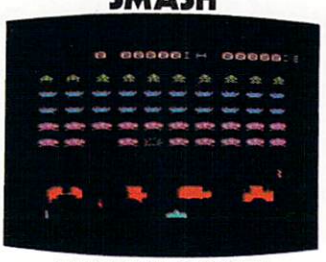


JUPITER LANDER

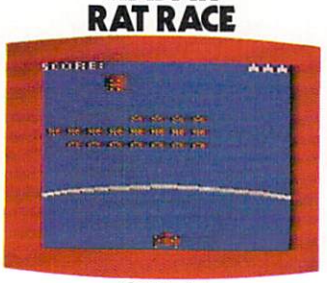
ER
EN



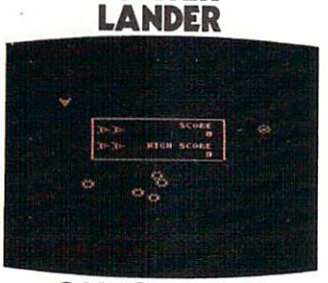
CHESS



VIC AVENGER



GORF



OMEGA RACE

What those extra few dollars get you is a simple little device called a Commodore VICMODEM.

It connects your telephone to your VIC 20™ or Commodore 64™ computer (resulting in something aptly called telecomputing), giving you access to information such as you see on the screens to your right.

Normally, you'd have to type a short program into your computer to help it make the final transition into a telecomputer.

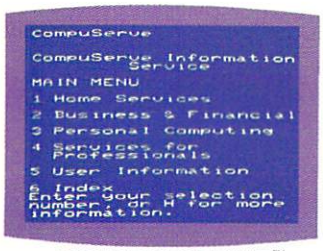
However, when you buy a VICMODEM, you'll find we've included a free software program. You just load it into your Commodore Datassette Recorder, and presto (give or take a moment or two), you have access to a vast library of information and games.

Speaking of free, Commodore also includes a free subscription and a free hour's time on CompuServe™ and Dow Jones News/ Retrieval Service,® a free trial offer on The Source,℠ and a discount program offer with Comp-U-Store and General Videotex Corp.

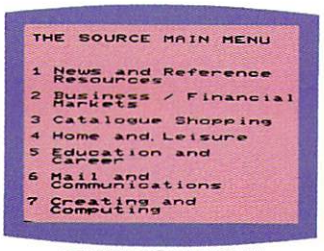
Let's see. Did we leave anything out? Oh, yes. Along with CompuServe comes a free membership in the Commodore Information Network. This is your HOTLINE to Commodore. (How often do you get to speak directly to a manufacturer?) Through it we

Certain offers subject to change. CompuServe is a trademark of CompuServe, Inc. and H.&R. Block Co. Dow Jones News/Retrieval Service is a registered trademark of Dow Jones & Co., Inc. The Source is a service mark of Source Telecomputing Corporation, a subsidiary of Reader's Digest Corporation, Inc.

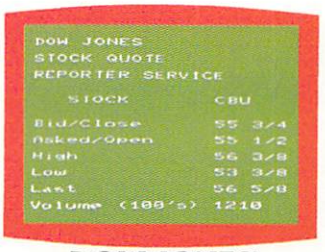
BUT DID YOU KNOW FOR ABOUT $100 YOU CAN ALSO GET IT TO DO ALL THIS?



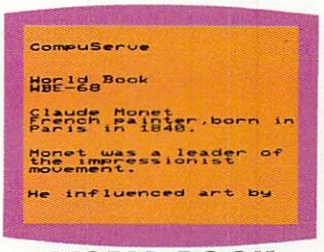
COMPU SERVE™



THE SOURCE℠

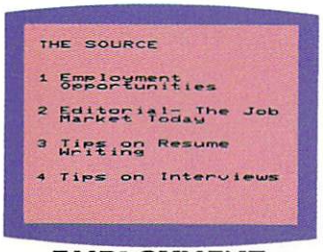


DOW JONES®

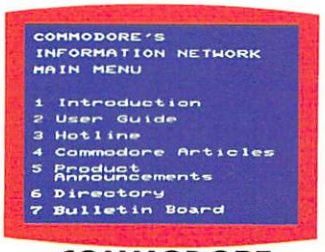


WORLD BOOK ENCYCLOPEDIA

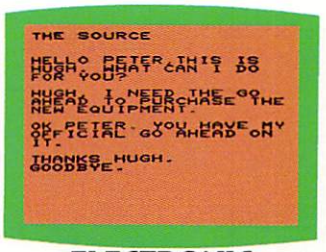
STOCK



EMPLOYMENT OPPORTUNITIES

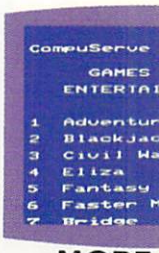


COMMODORE INFO. NETWORK

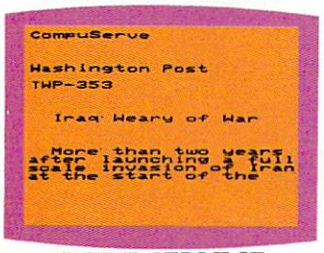


ELECTRONIC MAIL

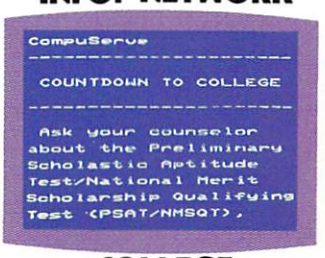
TRAVEL INFORMATION



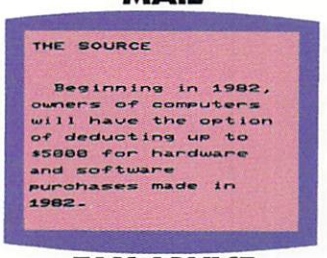
MORE



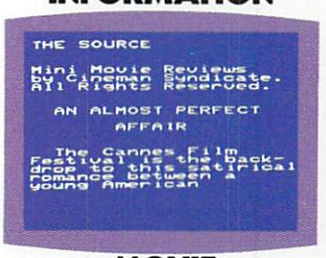
WIRE SERVICE NEWS



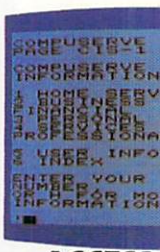
COLLEGE PLANNING



TAX ADVICE

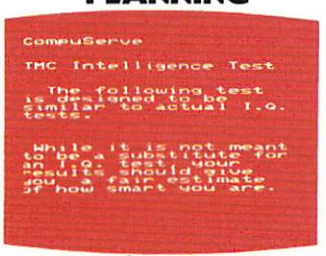


MOVIE REVIEWS

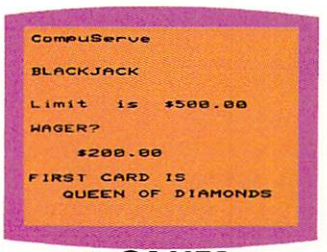


ACCESS COMP

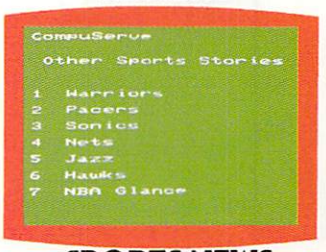
SHOP AT HOME



I.Q. TESTS



GAMES



SPORTS NEWS

BUSINE

can answer any questions you might have about your computer, or programming, or anything else Commodore-related, via electronic mail.

The Commodore Information Network is also your direct line to the Commodore Bulletin Board, which Commodore owners use to keep in touch with each other, for programming tips, Public Domain Software, and technical support.

THE COMMODORE VIC 20, A REAL COMPUTER FOR THE PRICE OF A TOY.

Altogether, these little extras we've included with our VICMODEM add up to a value of $197.50. A nice return on an investment of around $100.

Some computer companies think it's reasonable to ask as much as $500 for telecomputing capabilities such as ours.

However, with the Commodore VICMODEM selling for around $100, we feel we're being a whole lot more reasonable. Don't you agree?

commodore COMPUTER

These are just a few examples from our existing library of software available for the VIC 20. U.S.A.–P.O. Box 500, Conshohocken, PA 19428; Canada–3370 Pharmacy Avenue, Agincourt, Ontario, Canada M1W 2K4.

Commodore Information Network's Survival Kit

by Jeff Hand

This article is to help users navigate their way through Commodore's Information Network, and also become aware of the capabilities the Network provides. The purpose of the Commodore Information Network is to use the power of a national computer network (CompuServe) to give you the user a direct line to Commodore, disseminate information relevant to our computer users and give users a place to meet other users and exchange ideas and information. The Commodore Information Network can presently be divided into five sections: videotex, HOTLINE, bulletin board, databases, and conferencing. The Information Network has been very successful and well received by everyone.

VIDEOTEX

When you've connected into the Commodore Information Network, you have your choice of many different options. You can follow the menu structure and proceed through the videotex area. Or you could go directly to the page of videotex you wish to see (such as product announcements or the bulletin board), bypassing the menu structure. (This is especially useful for repeat users who have a specific videotex area in mind.) To navigate to a specific page, use the "GO" command: G CBM(X). (X) is the page number you want. Below is a listing of the page numbers in the videotex for direct access, and a diagram (page 22) of the categories of information available from the videotex main menu. You have your choice: use the menu structure provided, or with more experience go directly to any section.

Commodore Information Network

MAIN MENU

1 Introduction
2 Survival Kit
3 HOTLINE
4 Product Announcements
5 Bulletin Board
6 Commodore Magazine Articles
7 Directories
8 Commodore Tips
9 Price List
10 User Questionnaire

Enter selection or H for help:

DIRECT ACCESS PAGES

The following sections of the videotex area may be accessed directly by typing "G CBM-(page number)":

Main Menu CBM-1
Introduction CBM-4
Survival Kit Menu CBM-100
Hotline CBM-200
User Groups CBM-250
Commodore News CBM-300
Bulletin Board CBM-310
Commodore Magazine CBM-321
Power/Play Magazine CBM-322
Newsletter CBM-323
User Group Newsletter CBM-324
Product Announcements CBM-330
Directories' Menu CBM-400
Dealer Lists CBM-410
Educational Resources CBM-420
Commodore Tips CBM-500
Software Tips CBM-510
Price Lists CBM-750
Technical Tips CBM-800
Videotex Explanation CBM-1199
Bulletin Board Expl CBM-1200
Hotline Explanation CBM-1212
Database Explanation CBM-1216
Direct Access Pages CBM-1238

There are a few other navigational commands that are available to you. Below is a list of helpful commands available from the videotex area by typing "Help" at any (!) prompt. These commands are:

T—Top Menu Page (will place you into CompuServe's main menu)
M—Previous Menu (takes you back one menu page)
F—Forward a page
B—Back a page
H—Help (will give this listing)
R—Resend a page
S "n"—Scroll from item "n" (the text is formatted to your screen size rather than the normal 32 columns by 12 lines for videotex)
G "n"—Go directly to page "n"
N—Display next menu item
P—Display previous menu item

HOTLINE

One of the biggest advantages of the Commodore Information Network is that users can ask Commodore questions directly, and within a few days find the answer in the HOTLINE (0) database. To access the HOTLINE, simply type "G CBM-200" and you will be transported to the HOTLINE. After a brief explanation of the HOTLINE is given on your screen, simply start typing. There are no line prompts in the HOTLINE. The next business day Commodore will receive your question, determine an answer and upload the answer to the CompuServe system. You are then notified by EMAIL of the file name and the database section in which the answer is located, along with a short reminder on how to access files in the database. The answer file generally takes two working days before it appears in the hotline database. This is because CompuServe's public access area requires 24 hours for files to be processed and then an additional day for the file to be submitted to the database. (The HOTLINE on the videotex area differs from the HOTLINE database in that you leave questions on the Network HOTLINE and they are answered in the database HOTLINE and stored there for other people to look up later. Thus, many times people can save themselves the trouble of having to ask questions that have already been answered.)

Bulletin Board

One of the things you may wish to access often is the Commodore Information Network bulletin board. The bulletin board can be thought of as a corkboard for Commodore users to converse with each other. You can access the bulletin board by simply typing in GO CBM-310 at any prompt, or by using selection #5 from the main menu. Once you are connected to the bulletin board, you will be presented with the function menu. It will read as follows:

Commodore Business Machines

Function Menu:
1 (L) Leave a Message
2 (R) Read Messages
3 (RN) Read New Messages
4 (RM) Waiting Messages
5 (B) Read Bulletins
6 (CO) Online Conference
7 <Not used>
8 (MI) Membership Information
9 (OP) Change your Sig Options
0 (E) Exit from this SIG

You simply enter the number of your choice and press <RETURN>.

"Leave a Message" will let you leave a message on the bulletin board directed to a specific person or to everyone. Your message can be a question, a reply or maybe a request for information. The choice is yours. You will be prompted by the computer for the following information:

1. Name and User ID of the person you are sending the message to (unless it is to all)
2. Subject of Message

After entering the subject you will receive line numbered prompts for each message line. For example:

1:
2:
3:

A <RETURN> in the first column or a <CNTL> Z will end your message. Then you'll get the following menu:

Leave Options:

1 (S)	Store the message	(Put message on bulletin board)
2 (L)	List the message	(List message to verify editing)
3 (R)	Replace a Line	(Specify line, then retype text)
4 (D)	Delete a Line	(Specified line number is deleted)
5 (C)	Continue entering the text	(text will start on next line)
6 (A)	Abort the Leave option	(Cancels your message)

Enter selection or H for help:

"Read Messages" will let you read all the messages on the SIG. After this selection, the computer will inform you of the number of messages on the system and ask which messages you want to start reading. The format for messages in the SIG is as follows:

```
    : <zzz>                                        Sec. y
Sb: <xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
    DD-MM-YY                                   hh-mm-ss
Fm: <wwwwwwwwwwwwwwwwwwwwwwwwww>
To: <vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv>
```

z—The sequential message number assigned to this message.

y—The database section where the message is stored.

x—The subject header can be up to 24 characters long.

DD-MM-YY hh-mm-ss-the date and time when the message was stored. It is displayed in the reader's local time.

w—Writer of the message. This is provided by the SIG. It is the name that the user originally signed onto the Network with.

v—Recipient of the message. Either the user ID or the person's name or both are entered here. It is usually better to enter the user ID and avoid misspelling the recipient's name. This will insure that the message gets to the appropriate person.

Each message may be 96 lines or 2500 characters long (whichever comes first). The "Subject", "To" and "From" lines may each be up to 24 characters long.

After reading each message you will be presented with the following menu:

Read Option Menu:

1 (C)	Read next message
2 (RE)	Replay to current message
3 (T)	Return to the Function Menu
4 (D)	Delete this message

Enter selection or H for help:

This list of options is self explanatory. The letters in the parentheses are to remind you of the commands for command mode. More about command mode will be discussed later.

"Read New Messages" allows you to read all the messages put onto the system since the last time you accessed it.

"Read Waiting Messages" allows you to read messages that have been directed to you specifically.

"Read Bulletins" lets you read any announcements that the SysOp (system operator) has left for users in regard to the bulletin board.

"Online Conference": Another feature of the SIG is conferencing. Users may wish to schedule a live lecture or a discussion on a specific subject, or just plain gab away. Enter the conference (CO) feature from the function menu. Once you have done this, the screen will display your name and the channel you are on. Then just start typing! If you type "Help" at the beginning of any line you will get a list of conference commands, with brief explanations. These commands allow you greater flexibility when conferencing. (See the Conferencing section of this article.)

"Change Your SIG Options" allows more experienced users to tailor the SIG to their needs. Here is the menu that will appear on your screen with this option.

Selectable Options Menu

1	Change to command mode
2 (LL)	Change line length
3 (T)	Return to Function Menu
0 (P)	Make options permanent

By typing in "Help" after the above menu you can see what options are available to you. Below is a sample listing. The options with an asterisk (*) are your present settings for the bulletin board. Any option can be changed by entering the command at the selectable option prompt.

SIG User Selectable Options

T	—return to the top function level
ST	—stop between messages (*)
NS	—don't stop between messages
LL	—change line length (80)
BR	—set brief mode, which suppresses repetitious display of options
NB	—clear brief mode (*)
PC	—change prompt character
TWM	—type waiting messages
MWM	—mark waiting messages (*)
CN	—change name
DS	—set default login section (0)
RNT	—command types all messages (*)
P	—make options permanent
MEN	—use menus instead of command prompts

Any of the above commands can be used at the selectable option menu or when you're in command mode. Command mode can be accessed by choosing selection 1 on the user options menu. When you're in command mode there are no menus only a short prompt to remind you to put a command into the system. The prompts are:

Function:
Option:

That's it. So it's a good idea to keep a list of commands nearby if you're going to operate in the command mode. If you type in help at the command prompt you'll get a list of commands available to you as follows:

B	—bulletins	CO	—conference
D	—delete	E	—exit
G x	—go pg x	E(x)	—exit and do x
I	—instructions	L	—leave message
M	—prev. menu	NEW	—new changes to SIG
OP	—user options	QS	—quick scan
R	—retrieve	R x	—run SIG x
RT	—read thread	S	—scan headers
SD	—scan & display	SEN	—send message
SN	—section names	SO	—sysop function
SS	—set section	SU	—show usage
T	—go display	U	—user log
UST	—current users	V	—interests
X	—database	XA	—X ACCESS

To get a list of all the commands and capabilities available to you in the bulletin board type in the "I" command at the function prompt. This command will give a list of all the commands on the system along with explanations. The commands are in the following categories: message retrieval, formatting, scanning, leave, user changeable options, database sections, multiple commands, control keys, and other features.

Database Instructions

To access the Commodore Information Network database type "xa" at the function menu. Next you will be prompted which database you want to enter. When you see the "SIG/ACCESS:" prompt you know you're in the database area. The following ACCESS commands are available from our database.

PUB—retrieve from Public ACCESS
XA —change to new database
CAT—catalog
TYP—type a file
DOW—download a file
KEY—search keyword list
SUB—submit a file
DEL—delete a file
EXI —exit from ACCESS
HEL—explains ACCESS

1. PUB—The "PUBlic" command allows access to the Public ACCESS database, which is for the entire CompuServe system.
2. XA—The "XA" command allows you to change to a different SIG section database. You will be shown a list of the databases which you are authorized to access, and asked for the one you desire to use.
3. CAT—The "CATalog" command allows you to examine the contents of the public file system. The format is:

   CAT file. ext [User ID]/option/option....

Each visible file is listed in the following form:

NAME.EXT DD-MM-YY #

where "#" is the file size in bytes. If the file was submitted from your User ID with the /inv option, then the string "/inv" will follow the file.ext. In addition, your own files will be followed by the total number of times that each has been retrieved, and the most recent access date.

The CATalog is listed in User ID order, with the files in alphabetic sequence. The simplest form is:

CAT

which will give a brief list of all visible public files. The file.ext may have "wildcards" in them, where * in either the file or an ext position signifies any file will match the *. A "?" may be used to mean that any letter digit will match in that specific position. For example:

CAT *.BAS

will find any file with an extension of "BAS" in any User ID.

CAT ABC???.X?Z

will match any file whose name begins with ABC and whose extension contains an "X" as its first character, has anything in the middle, and ends in "Z." Also, any User ID will match the form:

CAT [User ID]

This will match any file submitted by the given user. Note that the square brackets "[ ]" are mandatory.

The options allowed are:

A. /age:n—output only if the entry has been SUBmitted within the last n days
B. /des—output the description of each file as given by the submitter.
C. /pause—stop after each file is listed; mostly useful with /des
D. /key:list—select only files which have the given set of keywords supplied by their submitter. The list may be a series of words separated by "+" to imply "or" or "&" to imply "and". Example:

   CAT/KEY:this+that&Basic

Will find files having keywords 'this' or 'that' and 'basic'

4. TYP—The "TYPe" command allows you to examine specific public files. The format is:

   TYP file.ext [User ID]

You are not permitted to TYPe files that you have placed into the Public ACCESS system. Entering a CONTROL P will abort the output and return to the "Command:" prompt.

5. DOW—The "DOWnload" command will transfer a public file into your own personal computer. The format is:

DOW file.ext [User ID]

You cannot DOWnload any files which you have SUBmitted to ACCESS. ACCESS will try to determine which software you are using to allow for the appropriate protocols.

6. KEY—The "KEY" command allows you to scan through the file KEYS [1,1]. This file is constructed from the keywords which have been supplied by users when they SUBmit visible files. The current keywords are displayed in sorted order, and are followed by the frequency of occurrence. The purpose of having these keywords is to provide a way to rapidly locate files of interest. If, for example, you were interested in BASIC programs for the VIC 20, you might use the following command:

CAT/KEY:BASIC + VIC 20

and you would only see those specific files.

Users are urged to check this KEY file before SUBmitting their own files and to choose keywords which accurately describe the file being submitted. The KEY command may be followed by a string of characters, in which case all keywords which BEGIN with the characters provided will be located in the KEY [1,1] file.

For example: KEY APP, will find APPLICATION, and APPROXIMATION (if these keywords are present).

7. SUB—Using the "SUBmit" command from a SIG serves to post keywords and a description of your Public ACCESS file into your SIG's database. All postings are assumed to be /VISible. When you SUBmit a file to a SIG database, that file must exist in the Public ACCESS database, and it will be copied into the SIG PREview file to await approval of the SYSOP.

8. DEL—The "DELete" command may be used to remove one of your SUBmitted files from ACCESS. It is entered simply as:

DEL file.ext

The file will typically be removed within 24 hours.

9. EXI—the "EXIt" command causes ACCESS to relinquish control. If you entered from the OK prompt, you will return there. If you entered from the SIG, you will return to the SIG.

HEL—The "HELp" command will list the SIG/ACCESS instructions.

Database Sections

0: HOTLINE—This database contains questions that have been left in the HOTLINE along with the answers. These files are stored permanently for future reference. If you have a question chances are it has already been answered in the HOTLINE database.

1: Manual Updates—Here you will find errata sheets for Commodore's manuals.

2: Vendors—Commodore has many outside vendors making and selling various products for Commodore computers. You can find a listing here.

3: Software—Free public domain software is available for all Commodore computers.

4: User Group Forum—User groups can use this area to post announcements, newsletters, and messages to groups.

5: Communications—Various bulletin boards across the nation are listed here. Reviews and evaluations of terminal packages and modems for Commodore machines will reside here.

6: Hardware—Information in this section will cover any hardware topic on Commodore computers.

7: Business—A report on Commodore's financial status is provided. Such things as earnings per share, total sales, and a financial outlook are here.

8: Baudy Bytes—This database is a compilation of books and magazine articles on Commodore equipment.

Sending Programs or Files to the Sysop

Users can submit programs or files to the appropriate database by using the filge (file generator) capability on CompuServe (refer to CompuServe's filge manual for a list of commands and capabilities). There are two ways that the file generator program can be used. First files can be uploaded from your microcomputer into a waiting open file, or second you can type your files or programs directly onto the system. Obviously, the first method is much easier so we'll cover it first.

You must be logged onto CompuServe and in to command (programmer's) mode. To get in to command mode type EXIT at the prompt. The command prompt will be an "OK". To create an open file type: FIL filename. CompuServe will return with "New file 'filename' created—ready". Your next step is to upload or type the program or file into the open filge file. When your file is complete use the "/ex" command to close the file. You can check your catalog with "CAT" to make sure your file is in your workspace.

Next go to CompuServe's Public ACCESS area by typing "R ACCESS". At the "ACCESS" prompt type "SUB filename[user ID]". It will take 24 hours for your file to appear on the system.

The next day go to the Commodore Information Network, access the database you wish to submit the file or program to and at the "SIG/ACCESS" prompt type "SUB filename[user ID]". Your submission will be brought to the attention of the Sysop the next time he accesses the system.

Conferencing
When you enter the conferencing mode, all you have to do is start typing and your message appears on everyone's screen that is monitoring that channel. Here are a list of commands you can use while in conference.

Prefix each command with a "/".
/TUN#—Tunes channel # (there are channels 1 to 36)
/MON 1,4—Listen to extra channels
/UNM 7,3—Unmonitor channels
/STA—Type channel status
/TIM—Type time, day and date
/UST—User STatus type out
/EXI—Exit conferencing
/WHO—Type PPN of last talker
/HAN—Change handle
/SCR xyz—Scramble on key "xyz"
/SMC xyz—Scramble and Monitor Clear
/XCL xyz—Xmit CLear
/UNS—UNScramble (both clear)
/SQU abc—SQUelch handle "abc"
/JOB—your job #
/HELP—Types out this message

SENd Function
This function is not in the conferencing mode but I think it deserves some attention. A user may wish to send another SIG user a short message. At the Function prompt enter the following:

SEN TxxYYY message

where xxYYY is a 2 digit number and a 3 letter code representing the user's present physical connection to CompuServe. The SENd command is useful to send another user a brief message even if he is doing something else at that time. For example, entering:

SEN TO1CLG Jack meet me on CB Ch 30 when you are done.

would display the following at the receiver's terminal.

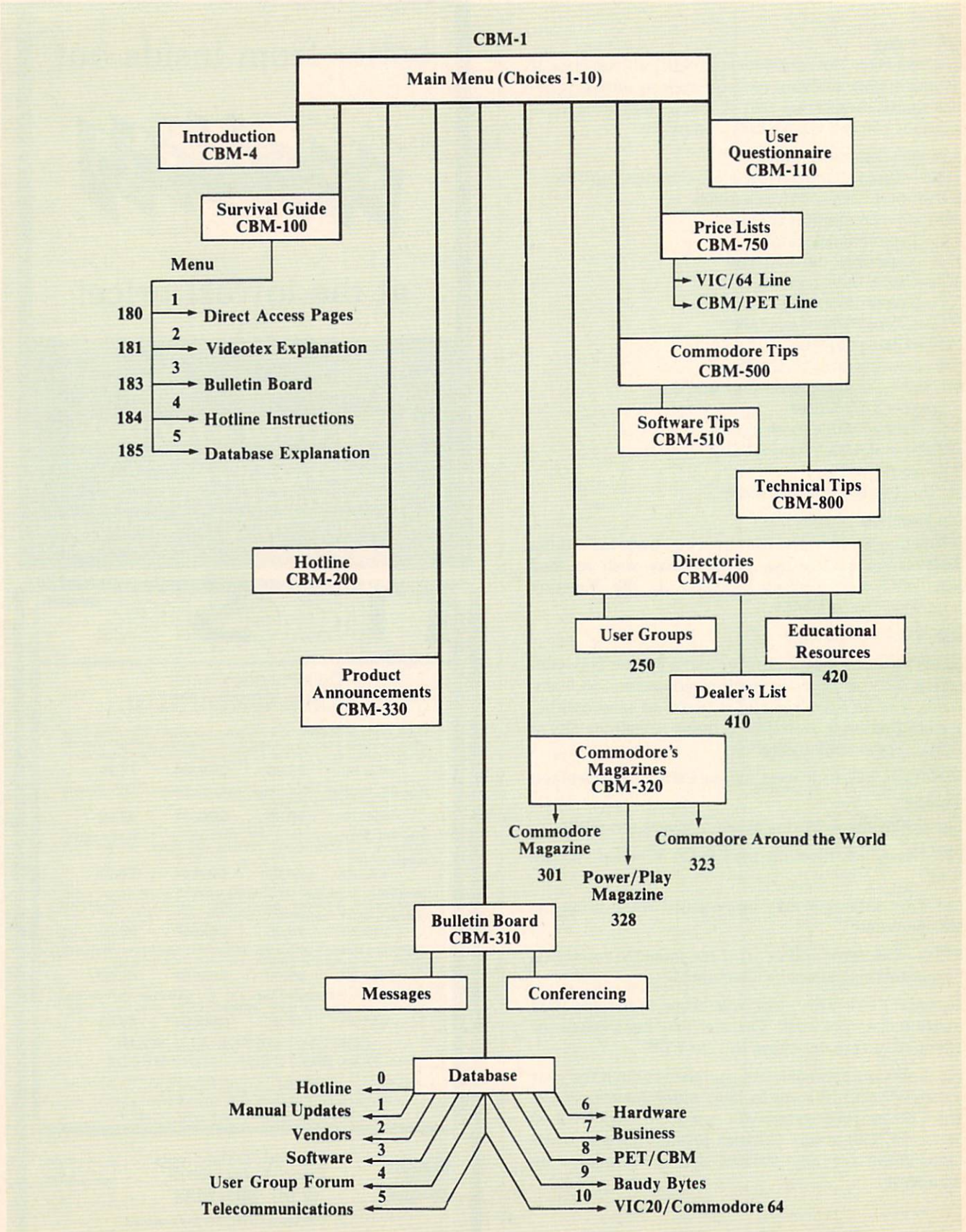
T23SFA:—Jack meet me on CB Ch 30 when you are done.

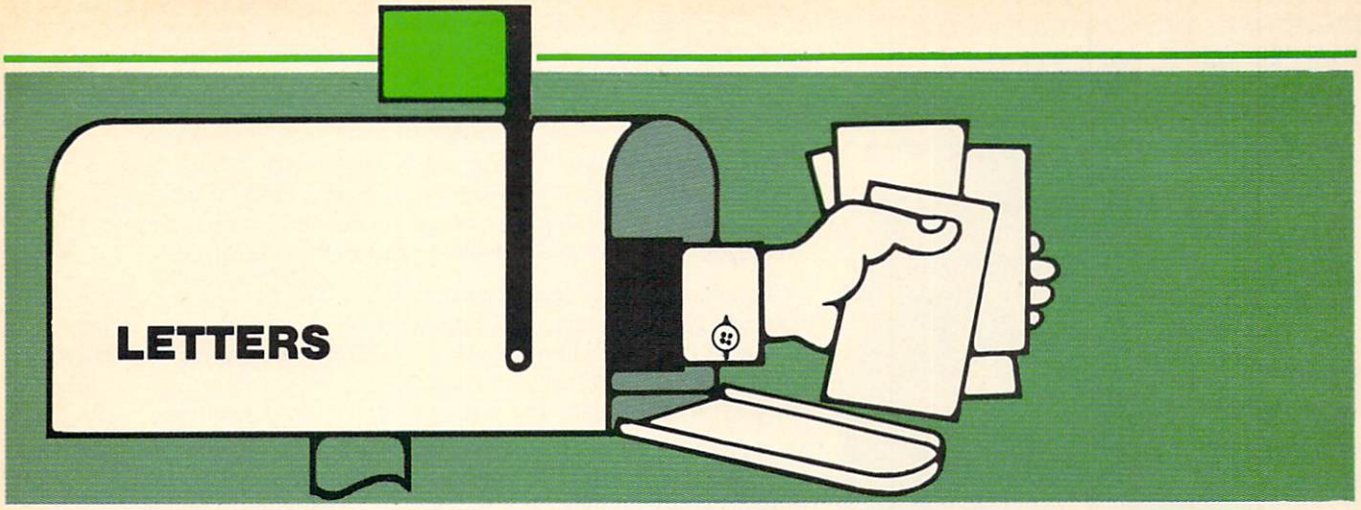
A few error messages may be returned while using the SENd command:

? BUSY—This error will occur if the intended receiver is in a program that cannot be interrupted.

? No such TTY—This error will occur if the specified connection does not exist. The user may have exited the S1 or you may have mis-typed the "xxYYY".

If this short user manual is not complete enough for your purposes or you just want to have more documentation on the CompuServe system you can go to page PCS 40 (type "G PCS 40 at any videotex prompt (!)) and order documentation for any services that are available on CompuServe.


CBM-1
Main Menu (Choices 1-10)
Introduction CBM-4
Survival Guide CBM-100
Menu
180 1 Direct Access Pages
181 2 Videotex Explanation
183 3 Bulletin Board
184 4 Hotline Instructions
185 5 Database Explanation
Hotline CBM-200
Product Announcements CBM-330
Bulletin Board CBM-310
Messages
Conferencing
Database
0 Hotline
1 Manual Updates
2 Vendors
3 Software
4 User Group Forum
5 Telecommunications
6 Hardware
7 Business
8 PET/CBM
9 Baudy Bytes
10 VIC20/Commodore 64
User Questionnaire CBM-110
Price Lists CBM-750
VIC/64 Line
CBM/PET Line
Commodore Tips CBM-500
Software Tips CBM-510
Technical Tips CBM-800
Directories CBM-400
User Groups 250
Educational Resources 420
Dealer's List 410
Commodore's Magazines CBM-320
Commodore Magazine 301
Power/Play Magazine 328
Commodore Around the World 323



October 10, 1982

Editor
Commodore Magazine
487 Devon Park Drive
Wayne, PA 19087

Dear Editor,

I really enjoyed Jim Butterfield's excellent article, "Machine Language Auto-Location," (*Commodore Magazine*, June/July 1982, pp. 82-84). I hope that you will continue to run more articles of this type.

To expedite the relocation process as described by Jim, I put together a small BASIC program. This program builds up the relocation code module as detailed in the article. For the benefit of the readers of *Commodore Magazine*, I have enclosed a listing of this program, which you may feel free to reprint.

The listing is fairly complete, in that the REM statements pretty much tell the whole story. I followed the protocol established by Jim in his article, so that any reader should be able to put it to use immediately. The only point not made explicit in the listing is the use of $BF (decimal 191) for the "end of the block" marker. This, of course, was the marker used in the original article.

The program was written on a CBM-8032, however it should be easily converted to all other Commodore computers. I hope this will be of some use to you and your readers!

Sincerely,

Thomas Henry

```
READY    ****************************************************************
         *                                                              *
         *                    A PROGRAM TO CREATE A                     *
100 REM  *                   RELOCATABLE CODE MODULE                    *
110 REM  *                      BY THOMAS HENRY                         *
120 REM  *                                                              *
130 REM  *            BASED ON THE ARTICLE BY JIM BUTTERFIELD           *
140 REM  *              "MACHINE LANGUAGE AUTO-LOCATION",               *
150 REM  *          COMMODORE MAGAZINE, JUNE/JULY, 1982, PP.82-84       *
160 REM  *                                                              *
170 REM  ****************************************************************
180 REM
190 REM
200 REM
210 REM  ***                  FOLLOW THESE DIRECTIONS:                  ***
220 REM  ***                                                            ***
230 REM  ***  [1]  ASSEMBLE ONE VERSION OF THE MACHINE LANGUAGE PROGRAM  ***
240 REM  ***       SO THAT ITS LAST BYTE ENDS UP AT $FFFF.  THEN, USING  ***
250 REM  ***       MICROMON OR SOME OTHER EXTENDED MONITOR, TRANSFER IT  ***
260 REM  ***       SO THAT IT SITS BETWEEN $6000 AND $7FFF.  (I.E., ITS  ***
270 REM  ***       LAST BYTE RESIDES AT $7FFF).                         ***
280 REM  ***                                                            ***
290 REM  ***  [2]  ASSEMBLE ANOTHER VERSION SO THAT ITS LAST BYTE ENDS  ***
300 REM  ***       UP AT $5FFF.                                         ***
310 REM  ***                                                            ***
320 REM  ***  [3]  LOAD IN THIS BASIC PROGRAM, THEN LOAD IN THE TWO     ***
330 REM  ***       MACHINE LANGUAGE MODULES, MENTIONED ABOVE.           ***
340 REM  ***                                                            ***
350 REM  ***  [4]  RUN THE PROGRAM.  A RELOCATABLE CODE MODULE WILL BE  ***
360 REM  ***       CREATED AND WILL THEN BE DEPOSITED IN MEMORY BETWEEN  ***
370 REM  ***       $2000 AND $3FFF.  (THE LAST BYTE WILL BE AT $3FFF).   ***
380 REM  ***                                                            ***
390 REM  ***  [5]  WHEN THE RELOCATION IS DONE, YOU WILL BE TAKEN INTO   ***
400 REM  ***       THE MONITOR AUTOMATICALLY WHERE YOU CAN SAVE THE NEW  ***
410 REM  ***       CODE.                                                ***
420 REM
430 REM
480 REM
490 REM
500 REM
510 PRINT"[CLEAR][2 DOWN][RVS]RELOCATING NOW..."
520 POKE52,0:POKE53,32:HI=32767:LO=24575:RE=16383:PC=0
530 HP=PEEK(HI-PC):LP=PEEK(LO-PC):PC=PC+1:IFHP=191THEN590
540 IFHP<>LPTHEN570
550 POKERE,HP:RE=RE-1:IFHP=0THENPOKERE,0:RE=RE-1
560 GOTO530
570 POKERE,0:RE=RE-1:POKERE,HP:RE=RE-1:POKERE,PEEK(HI-PC):RE=RE-1:PC=PC+1
580 GOTO530
590 D$="":B=1:RE=RE+1
600 I=INT(RE/16):J=RE-16*I:D$=CHR$(J+48-7*(J>9))+D$
610 B=B+1:RE=I:IFB<5THEN600
620 PRINT"[CLEAR]THE PROGRAM HAS NOW BEEN MODIFIED FOR RELOCATION."
630 PRINT"SAVE IT THROUGH THE MONITOR NOW.  WHAT NAME WOULD "
640 PRINT"LIKE TO SAVE THE RELOCATED CODE UNDER";:INPUTNA$
650 PRINT"[8 DOWN].S ";CHR$(34);"0:";NA$;CHR$(34);",08,";D$;",4000"
660 PRINT"[6 UP]":SYS54386
```

EDITOR'S NOTES

The world is moving from the Industrial Age into the Information Age so fast we've almost stopped noticing it. But the fact is, if you're living on earth (as I believe most of our users are), you're inevitably involved—whether you want to be or not.

What are the new age's characteristics? Mainly faster, cheaper communication of more different types of information over longer distances than ever before. And you, as Commodore users, have the capabilities—literally at your fingertips—to be not just involved but actually in the vanguard of this communications revolution.

In this issue we take a look at the phenomenal developments in communications as they apply to Commodore users. If you ever thought your computer was just a fancy typewriter or an elaborate calculator, you have some surprises in store for you. If you've known all along that you can access enormous databases, but weren't sure what they were or how to get to them, you'll find at least the basics of what you need to know here. (According to Jeff Hand, our telecommunications expert in residence, it would take a veritable encyclopedia to list *everything* that's available—a project he plans to actually undertake sooner or later. In his spare time. But more on that later.)

I think it's time to mention that this year promises to be another big one for Commodore users, with more and better hardware, software and support available to them. That's great news. However, because of the speed at which Commodore's technology is moving forward, it's sometimes difficult for our users to keep up with the changes. As a result we get many letters and phone calls from people who need information on the latest developments.

We try to answer all your questions individually, but sometimes you need more detail than we can provide in a letter or short phone conversation. So I thought this would be the ideal time, being a new year and all, to remind you that there are many people out there—maybe right in your town—who use Commodore equipment, and who DO have the time to talk to you about your questions and concerns. You can find these angels of mercy in our user groups.

User groups offer an invaluable service that we as a company simply cannot provide—personalized, individualized attention to your interests and problems, whether you're an experienced programmer or a bumbling novice. That's why we're so committed to keeping our list of user groups up-to-date, and so anxious to post fledgling groups—and any other user-to-user notices—on our user bulletin board. We want our users to find each other, because we know how important individual contact is in this field.

Say, for instance, you're having a problem getting a program to run after you've spent a couple of hours typing it in, and have proofread it until you've gone cross-eyed. Where can you turn? Why, to your local user group. Somebody's bound to spot the problem.

Got a question about assembly language, subroutines, function keys, or a new product? You're probably not alone—and there's somebody in your local user group, or a group near by, who can answer those questions. Better yet, maybe you're the person who knows the answers, and are longing for the opportunity to share what you know.

We presently list about 170 user groups (see page 102), many of whom produce monthly newsletters. The groups on our list are located mainly in the United States and Canada, but we know many more groups exist around the world who simply haven't let us know, yet. If you're one of those, why hesitate? Or, if you've been wishing there were a group in your area, why not start one? It's easy if you post a notice on our user bulletin board, here or on the Commodore Information Network. And the results may surprise you. There are Commodore users hiding, it seems, in some of the most unexpected places.

So take advantage of the fact that we have more computers out there, worldwide, than anybody else, and get in touch with other Commodore users. We think you'll enjoy the contact—our groups seem to be pretty sociable, from what we've seen—and learn a lot, too.

Meanwhile, welcome to the Information Age, the Communications Revolution, and the first 1983 issue of *Commodore Magazine*.

Diane LeBold
Editor

FROM WHERE I SIT

by David Rosenwald
Director of Education Marketing

Because Commodore has traditionally supported excellence in education through the use of technology, and produced innovative, user-friendly computers at very affordable prices, we have been fortunate to become a leader in the educational market. However, we know we cannot maintain our reputation for excellence by simply relying on the past. That is why we are constantly renewing our commitment to education with new, better, more cost effective hardware, increasingly sophisticated software, and broader, more effective support programs.

Recently we have become more deeply committed than ever to increasing service and support to our educational users. As a result, we have established several specific methods for supporting the educational community. The major parts of our support program are: 1) a regional network of education specialists who are familiar with both Commodore equipment AND the concerns of educators 2) a restructured headquarters staff 3) special discounts on equipment for qualifying schools 4) education resource centers, through which educators help educators 5) regional seminars for teachers and 6) increased participation in state, regional and national educators' conferences.

Our regional education specialists, all of whom are former teachers, work with educators and Commodore dealers to make sure questions are answered and needs fulfilled. They are also responsible for supporting teacher seminars in their respective regions. Please feel free to contact the specialist in your area:

Midwest:
Mrs. Terry Anders
Commodore Business Machines
2246 N. Palmer Drive
Schaumburg, IL 60915
312-397-0075

South:
Mrs. Pam Spillios
Commodore Business Machines
4350 Beltwood Parkway S.
Dallas, TX 75234
214-458-1000, ext. 220

East:
Ms. Pat Walkington
Commodore Business Machines
761 Fifth Avenue
King of Prussia, PA 19406
215-666-6800

West:
Mr. Jim Bussey
3824 Cougar Place
Modesto, CA 95356
209-526-0223

In addition, working with me at headquarters are Dr. Dan Kunz, Program Manager for Educational Systems, Cindy Doms, communications and administrative coordinator, and Emma Jean Mungin, our secretary. We can all be reached at 215-687-9880.

Our many education resource centers continue to serve as a primary grassroots support system for educators, since it is here that direct contacts, teacher-to-teacher, are made. The information exchanged through our resource centers has proved to be invaluable for teachers working with computers, many for the first time, and many on limited budgets and tight time schedules. A school or intermediate unit may, with the approval of their school board, apply to Commodore to become an education resource center. If you would like to find out more about this program, write to the Education Resource Center Coordinator, Commodore Business Machines, 487 Devon Park Drive, Wayne, PA 19087.

In addition to increasing and improving support programs for education, Commodore will also continue to increase the amount and quality of software available to educators. In fact, we hope to be able to provide, through large educational publishers, software and coordinated texts in many subject areas.

We see educational software as having three different uses: academic, business education and administration. In the academic area, we recently made available—in the public domain—656 programs designed to run on most Commodore computers. These programs are teacher-developed and address specific classroom needs. Each program is catalogued by topic, grade level and degree of student/teacher participation. Also included as part of that package are programs for administration—grade reporting, attendance, statistical analysis and similar functions.

This is just the beginning of what will continue to be an unprecedented commitment to educational software. Other recent announcements include the soon-to-be-available LOGO and PILOT software for the Commodore 64. In addition, a PET emulator has been developed that allows the 64 to use over 90% of the software existing

for the PET series computers.

In the area of business education, we have found that the best way to learn a business application is to actually use the software used in industry. For instance there seems to be little advantage to only learning ABOUT word processing. To gain the greatest benefit students need to USE a word processor—or electronic spreadsheet. We at Commodore continually try to make such products available to our educational users.

Of course, Commodore is first and foremost a leader in the production of advanced computer hardware. Although we do many things well, this is the backbone of our reputation for excellence.

Our recent developments are no exceptions to that well established rule. The Commodore 64, for example, with its high resolution color graphics, music synthesizer and extraordinary memory capabilities—all at an affordable price—will continue to be an outstanding tool for educators for many years to come. The 64 is even more cost effective when it is networked in the classroom, or when it is connected to any of the major telecommunications data bases with our very inexpensive modem. The availability of sprite graphics further enhances the learning impact of the 64.

For more advanced educational applications—most specifically for teaching programming in the whole spectrum of computer languages—the SuperPET is the logical choice. In fact, there is no other microcomputer on the market with built-in language capabilities of the SuperPET. With 96K, an 80-column screen and APL, FORTRAN, BASIC, PASCAL and COBOL languages, as well as telecommunications capabilities, this microcomputer is ideal for sophisticated school applications. The SuperPET is an unexcelled development tool, and it serves business and other educational purposes, as well.

Commodore has always played a major role in education. As we continue to excell in hardware, software and support, we will be better able to meet the present needs—and anticipate the future needs—of our educational users, so we can better live up to our motto: "Commodore—committed to excellence in education through technology." C=

Updated Commodore Software Encyclopedia Soon Available as "Commodore Resource Encyclopedia"

If you haven't seen it yet, you will be seeing it soon—the most comprehensive listing ever collected of products and services for Commodore computers, including offerings for the Commodore 64. With the addition of several new categories, including "Communications," "Independent Programmers and Consultants," and listings of authorized Commodore dealers and Commodore user groups, the new edition is no longer simply a software encyclopedia. For that reason, this third edition—all 600 pages of it—has been renamed the *Commodore Resource Encyclopedia*.

The new *Encyclopedia* lists over 1900 products and services—almost doubling the number in the last edition. In addition to expanded software listings, the new "Communications" category provides a listing of hundreds of databases in the United States and Canada. These databases range from large national networks like CompuServe, THE SOURCE and Dow Jones News/Retrieval down to single-computer bulletin boards that might be based in a user's livingroom. All are accessible using Commodore equipment.

Under "Independent Programmers and Consultants" Commodore users can find a list of over 50 people who customize or modify software packages to better suit individual needs, who can write a customized application from scratch, or who can recommend the best Commodore systems for specific applications.

Be sure to look for the new *Commodore Resource Encyclopedia* at your authorized dealer soon. C=

New Plotter Printer for VIC 20 and Commodore 64 Scheduled for Early this Year

Commodore's 1520 plotter printer, designed for use with the VIC 20 and Commodore 64, will be appearing early this year. The 1520 is an x-y four-color plotter that uses ball point pens as writing instruments. Capable of high accuracy plotting with .2mm resolution, it is an excellent tool for creating graphs, charts and graphic designs, as well as four-color text.

The 1520 can be plugged directly into the computer's serial interface port or can be daisy chained with as many as four disk drives. It prints 80 characters per line in a variety of sizes, and can also print characters on their sides, a real convenience when creating charts and graphs. It uses standard 4½ inch rolled adding machine paper. C=

Commodore Introduces "A Computer Desk Built by the Computer Company"

Commodore will soon be producing a new computer desk capable of holding a complete Commodore system—computer, printer and disk-drive—at an affordable price.

The arborite desktop is typewriter height, measures 26" x 48", and is 1⅛" thick. It has heavy steel legs and frame, with adjustable leveling glides for easy set-up. The disk compartment is ventilated, and has handy cable slots for convenient connection to other equipment. C=

Commodore Imports VIC 20 Applications Programs

A wave of programs is on its way from Commodore in England, scheduled for delivery during the first half of the year. It includes the first "serious" applications software released for the VIC 20, including a word processor and a spreadsheet program, as well as several general interest programs.

The applications programs are set to sell for under $100.

SimpliCalc is an electronic spreadsheet program. This program has many features included in more expensive financial planning programs.

VIC File is a data management system for the VIC with 16K extra RAM and a 1541 disk drive. It lets the user easily set up files and cross-referencing systems.

VIC Writer is a sophisticated word processing system, suitable for manuscripts and correspondence.

A series of general interest programs is to be packaged like paperback books, stressing themes like self-improvement and education as well as novelty items. The first releases are Know Your Personality, Know Your I.Q., Know Your Child's I.Q., Robert Carrier's Menu Planner, and Quizmaster. All programs include a booklet and computer tape cassettes and were developed for Commodore by recognized experts in their field. C=

Commodore Customer Support Team

T.J. Rizol, Jr.
Manager/Customer Support

It is a well known fact that Commodore computers are the best computers for the money. A major influence in the decision-making process in buying a computer, however, is the support of the product. The intention of this article is to cover this important issue.

The Customer Support Group is situated here at the headquarters in Wayne, Pennsylvania. The charter of the group is to be a central source of information to the end user whether it be by telephone, mail, or telecommunications network.

Complete support of the customer or potential customer involves teamwork. The dealer plays an important role in this and should be the first place of contact. In many cases the Support Team will aid users in locating a dealer in their area or direct them to the appropriate channel. Persons interested in placing an order for hardware, software, and manuals are naturally directed to a local dealer. It is beyond the parameters of the group to have knowledge of dealer inventories, thus several dealer names are often given. Availability information on products and software can be obtained from the group but exact availability should be obtained through our dealer base.

Inquiries of all varieties are handled by our staff. From literature requests to technical questions, on the VIC 20 to the SuperPET. Documents and write-ups are available for mailing to those customers who find certain functions of the computer unclear. The main theme here is time-efficient support. Troubleshooting and debugging of customer programs is not possible for this reason. Highly technical questions may require consultation with the technical staff and therefore take a short time to respond to. In these circumstances a detailed letter is helpful to us.

Commodore's most recent commitment to support is by way of the Commodore Information Network. This service is incorporated in the CompuServe telecommunications network.

Accessed with the use of a modem this is a fun, easy, and inexpensive way to obtain information on products and answers to questions. Of particular note is the HOTLINE portion of this network. This section is operated by the Customer Support Group as a question and answer service. Users simply pose their questions on the HOTLINE which automatically lists the user ID number. On the pursuant day, upon accessing the service, the users find a message in their Electronic Mailbox (EMAIL) telling them where in the section to find the answer supplied by the Support Team. This is an exciting way to achieve answers to questions. For a general overview of the CompuServe network reference the articles on telecommunications appearing in this issue.

The Customer Support Group can be reached between the hours of 9:00 A.M. and 8:00 P.M. EST. The phone number is (215) 687-4311.

The mailing address is:

Commodore Customer Support
487 Devon Park Dr.
Wayne, PA 19087

Complete instructions and phone number listings for the Commodore Information Network are included in the package which accompanies the VICMODEM. If another type of modem is used call CompuServe at 1-800-848-8990 to obtain the correct phone number in your area.

Commodore's commitment to end users is ongoing. This is the first in a series of articles on the Customer Support Group. Updates and specifics such as most often asked questions, and technical write-ups will be covered in the future.

Gortek Is Coming

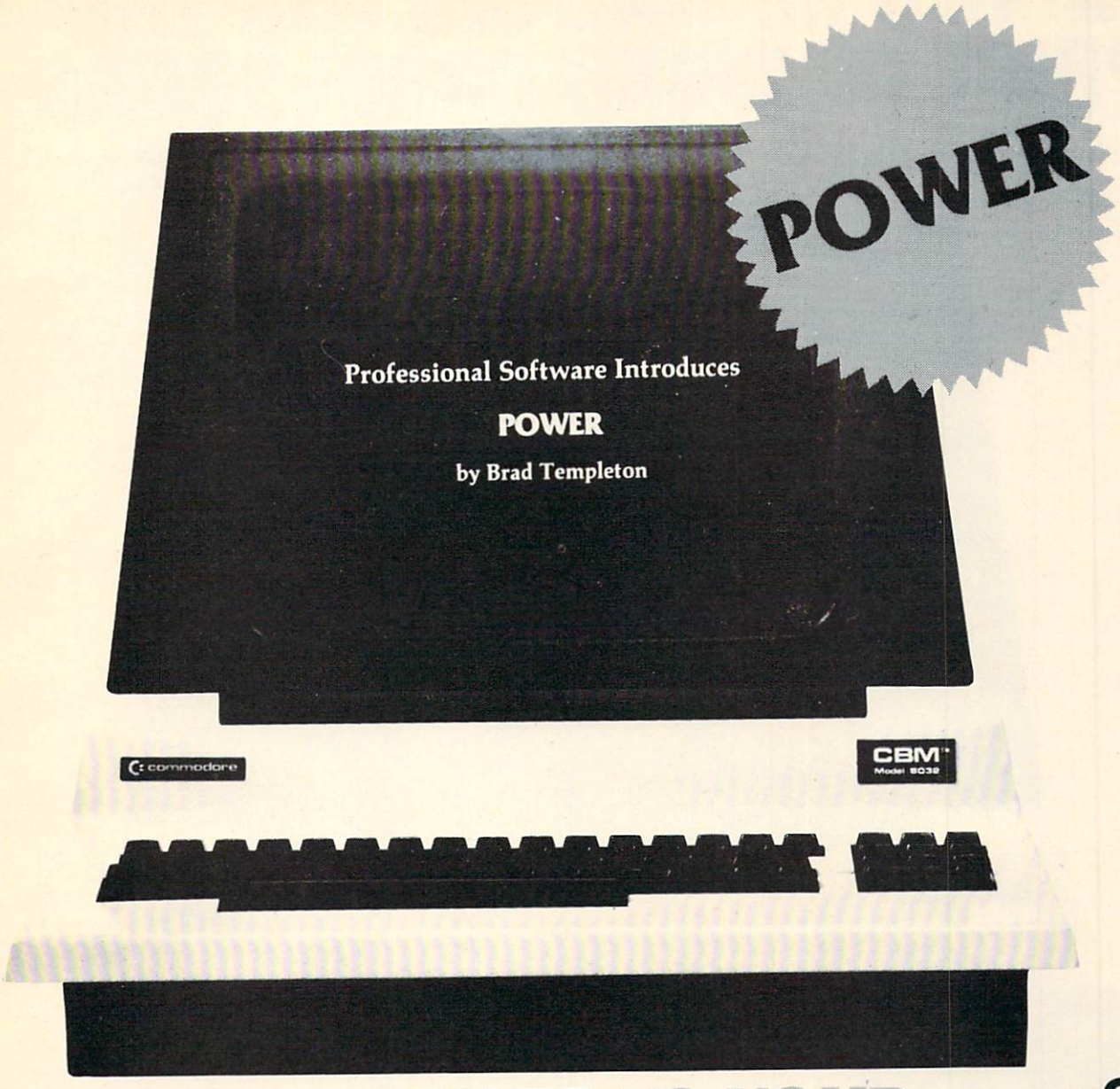
Gortek is the main character in a new series of space adventure stories designed to teach BASIC programming. The first lesson consists of two cassette tapes containing 12 educational programs and a book, and is called *Gortek and the Microchips*™. The book includes imaginative full-color illustrations, large easy-to-read type, and is written so that it may be read by older children or used by younger children with parental assistance.

The unique combination of the storybook and computer lessons makes Gortek a fun experience for adults as well as children.

As the story goes, the planet Syntax is being invaded by the fearsome Zitrons. Gortek works furiously to teach the microchips to program the computer to repel the attack. The "Microchips Training Manual" teaches the child how to stop the Zitrons—by learning to program the computer!

Those who complete the lessons and defeat the Zitrons earn the right to wear the Gortek badge which comes in the package.

This innovative approach to computer education was developed by three English school teachers who wanted to make programming fun to learn.



ADD POWER TO YOUR COMMODORE COMPUTER

$89.95

POWER produces a dramatic improvement in the ease of editing BASIC on Commodore's computers. POWER is a programmer's utility package (in a 4K ROM) that contains a series of new commands and utilities which are added to the Screen Editor and the BASIC Interpreter. Designed for the CBM BASIC user, POWER contains special editing, programming, and software debugging tools not found in any other microcomputer BASIC. POWER is easy to use and is sold complete with a full operator's manual written by Jim Butterfield.

POWER's special keyboard 'instant action' features and additional commands make up for, and go beyond the limitations of CBM BASIC. The added features include auto line numbering, tracing, single stepping through programs, line renumbering, and definition of keys as BASIC keywords. POWER even includes new "stick-on" keycap labels. The cursor movement keys are enhanced by the addition of auto-repeat and text searching functions are added to help ease program modification. Cursor UP and cursor DOWN produce **previous** and next lines of source code. COMPLETE BASIC program listings in memory can be displayed on the screen and scrolled in either direction. POWER is a must for every serious CBM™ user.

Call us today, for the name of the Professional Software dealer nearest you.

Professional Software Inc.
51 Fremont Street
Needham, MA 02194
Tel: (617) 444-5224 Telex #951579

CBM is a trademark of Commodore Business Machines, Inc.

IF YOU'RE WAITING FOR THE PRICE OF WORD PROCESSORS TO FALL WITHIN REASON,

Everyone expected it would happen sooner or later...with WordPro PLUS™ it already has! Now all the marvelous benefits of expensive and advanced word processing systems are available on Commodore computers, America's largest selling computer line. WordPro PLUS, when combined with the new 80 column CBM™ 8032, creates a word processing system comparable to virtually any other top quality word processor available—but at savings of thousands of dollars!

New, low cost computer technology is now available at a fraction of what you would expect to pay. This technology allowed Commodore to introduce the new and revolutionary CBM 8032 Computer.

WordPro PLUS turns this new CBM 8032 Computer into a sophisticated, time saving word processing tool. With WordPro PLUS, documents are displayed on the computer's screen. Editing and last minute revisions are simple and easy. No more lengthy re-typing sessions. Letters and documents are easily re-called from memory storage for editing or printing with final drafts printed perfectly at over five hundred words per minute!

Our nationwide team of professional dealers will show you how your office will benefit by using WordPro PLUS. At a price far less than you realize.

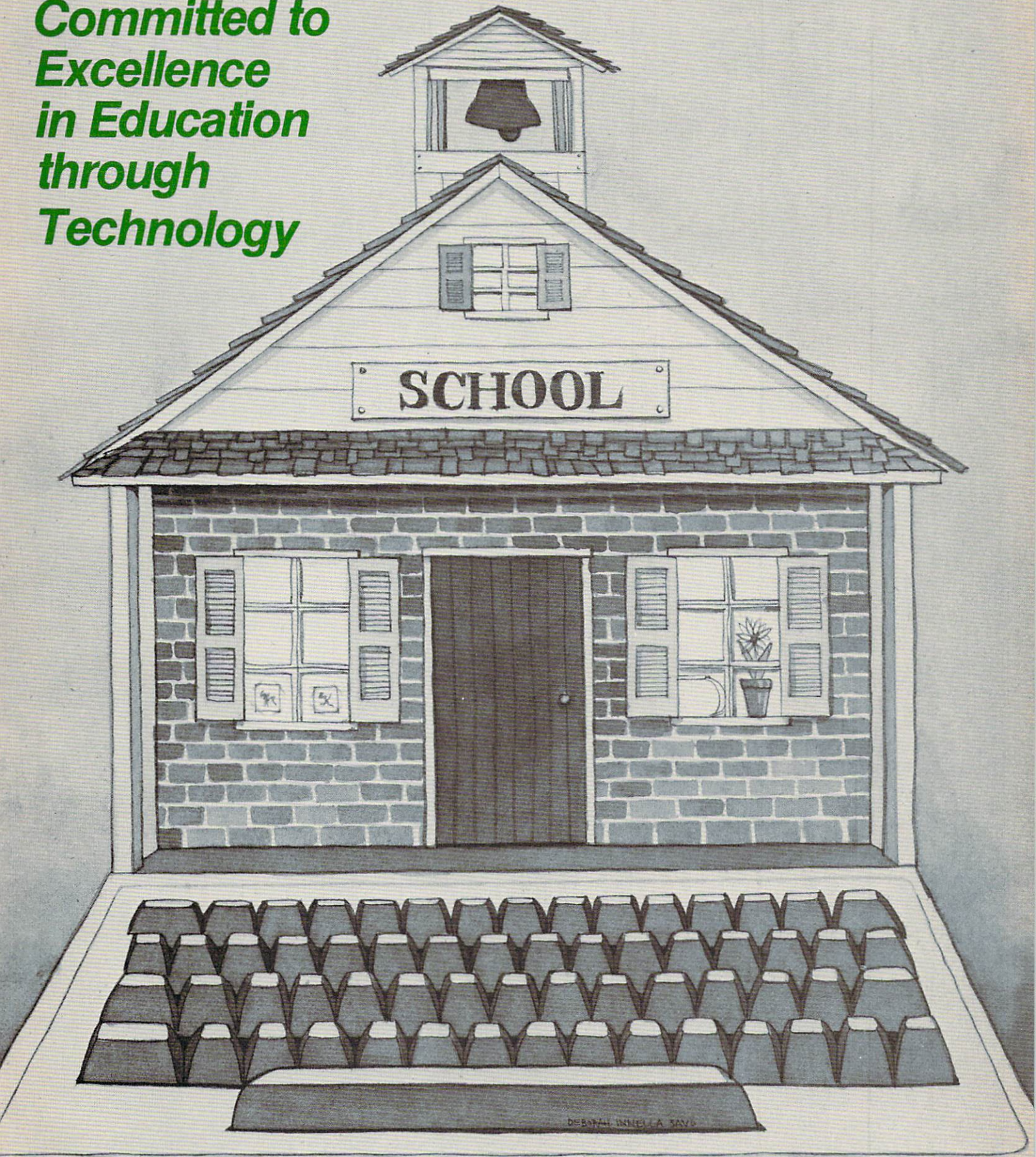
Invest in your office's future...
Invest in WordPro PLUS...
Call us today for the name of the WordPro PLUS dealer nearest you.

Professional Software Inc.

51 Fremont Street
Needham, MA 02194
Tel: (617) 444-5224
Telex #951579

CBM is a trademark of Commodore Business Machines, Inc.

Commodore: Committed to Excellence in Education through Technology



What do people think about Computers?

by Doris Dickenson
Pine Crest School, Sebastopol, California

As part of our fourth-grade computer literacy curriculum (inspired by having our Commodore 64 prize computer in the classroom) the students did a computer survey among their friends, teachers and relatives to find out what they thought about computers. The activity of the survey brought some real meaning to the time we were spending on the role of computers in our lives today.

Following our discussion about the uses of computers in our community, the children suggested questions suitable for use in a short survey. All of the eight questions we decided on could be answered either "yes" or "no." Only one question required some additional information to be written in. It was suggested that the students ask for help if they could not write down the response.

Taking part in doing the survey was voluntary, but soon even the less enthusiastic students volunteered to help. Each one was given a small, but official-looking, Computer Reporter card, identifying him or her as part of the computer literacy class. In class, the students practiced introducing themselves and giving the survey. Some of the school personnel were assigned by lot to be surveyed so they wouldn't be deluged by "reporters."

Because the class meets only once a week, several weeks were allowed for the questioning, with lots of encouragement and additional forms available during that time. As an added incentive, each student who turned in at least one completed survey was given a star on his or her classroom "reward card."

To count the survey results when they were completed, the class was divided into groups of three students; one student being the "reader" and the other two tallying, as a check on each other. Since the results were recorded on survey forms, the "reader" simply stated "yes" or "no" for each question as the other two tallied. It was then easy for me to count their totals for a final tally. We averaged almost a hundred responses.

We followed the survey with a discussion of the implications of our results. We noticed how popular computers seemed, even though few people knew how to run one, and even fewer owned one. Of course, the children supported the popular idea of computer instruction in schools. Since tallying the survey coincided with the meeting of our District Computer Committee, the results were reported to them, which made the survey seem doubly important to the students.

The results of our survey were as follows:

The person answering is
35% under 18 yrs. 65% over 18 yrs.

1. Do you like computers?
   88% Yes 12% No Why or why not? (See below)
2. Do you know how to run a computer?
   32% Yes 68% No
3. Do you own your own commputer?
   5% Yes 95% No
4. Would you like to learn how to use a computer?
   88% Yes 12% No
5. Is a computer used where you work?
   32% Yes 33% No 35% Don't work
6. Do you or your children have a computer in their school?
   68% Yes 32% No Note *
7. Do you think we should teach children in our schools how to use computers?
   93% Yes 7% No
8. Do you think computers will replace you in your job?
   19% Yes 81% No

**We felt that our single classroom computer, the only one in the district, may have accounted for a large number of these votes, even though it is not part of a district program and most children are not exposed to it.*

Some of the more common responses about why a person liked computers were:

They are fun to play with
They can be helpful
They are efficient and interesting
They make life easier
They are important for the future

Some of the responses about why a person didn't like them were:

They invade privacy
They make a lot of mistakes
They are not interesting to me

One of the most rewarding things of all was an eloquent letter from a parent who felt computers should be placed in a subordinate role to traditional and humanistic endeavors, and felt an inordinate amount of significance is currently being placed on their importance. He felt that computers were undeniably the wave of the future and were useful as a tool for efficiency and speed, but that as a means to self-fulfillment, emotional well-being, or an adjunct to solving the problems of our world, they had their limitations. Needless to say, this letter, too, was presented to the District Computer Committee. C=

These are the first two parts of a four-part Commodore 64 user guide for children, written by fourth-grade teacher Doris Dickenson.

YOU AND YOUR COMPUTER

A guide for students using the Commodore 64 computer

Part I:

Introduction to the Keyboard

If you follow these step-by-step directions, with a little practice you will find many interesting things that you can do with your computer. You may notice that your computer looks like a typewriter to you because the keyboards are somewhat alike. We are going to find out some of the ways that a computer keyboard is special.

*If you do not see the red "on" light at the top right-hand corner of the keyboard, flip the "on/off" switch or ask an instructor to help you. Also make sure your TV or monitor is on. You should see the word READY and a blinking square, or *cursor,* on your screen.

A. Numbers

The top row of your computer has the numbers from 1 to 0. You are going to practice typing numbers.

Practice:

1. Type in any number
2. Press the RETURN key.

Question:

Does a typewriter have a number "1" like your computer?

*If you see the message "?Syntax error" ignore it at this time as long as it is followed by READY and the blinking cursor.

3. Type in any other number.
4. Press the RETURN key.
5. Type in other numbers, pressing the RETURN key after each number.

*At *any* time, when you want to clear the screen of all your typing, press the SHIFT and CLR/HOME keys at the same time.

B. Letters

The keyboard letters are in the same position as on a typewriter. Any practice you can get on a typewriter will help you locate keys on your computer.

Practice:

(Clear your screen before you begin.)

1. Type your name
2. Press RETURN

*At this time all your letters will be in capitals. We will learn how to change that later on.

3. Now type in several sentences. If you need to use a period, you will find it in the bottom row.

Questions:

What does the computer do when it fills up one line? Does it follow the proper rules for dividing words?

C. Making corrections

Practice:

1. Type your name and make a mistake in spelling it.
2. Hold down either SHIFT key and press CRSR (cursor) key. This will move the cursor to the left. Move it until the flashing cursor is on top of the letter you want to change. *Now* type in the correct letter and press RETURN.
3. Press the CRSR key *without* pressing the SHIFT key.

Questions:

What do you do to move the cursor to the left? What do you do to move it to the right? What happens when you *continue* to move the cursor to the left past the edge of the screen? To the right past the edge?

4. Press SHIFT and CRSR . Press CRSR without the SHIFT.

Questions:
What do you do to move the cursor up? What do you do to move the cursor down?

5. Type your name and put in an extra letter. Use SHIFT and CRSR to go back until the cursor is *one letter to the right* of the extra letter. Press INST/DEL key on the top row, then RETURN. What happened? (DEL means delete or remove.)
6. Type your name and leave out one letter. Move the cursor back until it is on the letter *to the right* of where you want to add a space. Press SHIFT and the INST/DEL key *quickly.* (INST means insert or put in.) Add the missing letter.

Question:
What happens when you *hold down* the INST/DEL key?

D. Using colors

The CTRL (control) key lets you set colors as well as do other special things.

Practice:

1. Press CTRL and 9 at the same time and release them. Nothing seems to have happened, but now hold down the SPACE BAR, the long bar at the bottom of the keyboard. When you have had enough of that, try typing your name or some other words. You have actually been using RVS ON (reverse on) that you see on the front of the 9 key.
2. Press CTRL and 0 (zero) and see what happens when you hold down the SPACE BAR and when you type letters.

Question:
What do you think RVS OFF on the front of the 0 (zero) means?

3. Press CTRL and 9 (to reverse the bars) and then press CTRL and 8 and release them. Press the SPACE BAR. The bar on the screen should be yellow. Press CTRL and any other number keys. The colors are printed on the front of the number keys. Have your instructor help set the display controls on the TV if necessary.
   Practice with these basic 8 colors.
4. Now press C= and the different number keys again. You now have 8 more colors to use.
5. Restore your system to normal by pressing C= and 7 together, then RETURN, then SHIFT and CLR/HOME.
6. Press CTRL and any color key (without pressing 9 first). Type your name. Change colors by pressing CTRL and any other color key.
7. Now try this with the C= key and any color key. Type some more words.
   Try making different color patterns.
8. Restore the system to normal. (See 5 above if you forgot how.)

E. Graphic symbols

By using certain keys you can get graphic (or picture) symbols instead of numbers or letters. These symbols are marked on the front of each key instead of on the top.

Practice:

1. Press the SHIFT key and *continue* to hold it down while you press any key with a graphic symbol.
2. Continue to hold down the SHIFT key and press the key you chose several more times.
3. Continue to hold down SHIFT and try the other symbol keys.

Question:
Does holding down SHIFT and pressing a symbol key cause the right or left symbol on the key to be printed?

4. Press C= and press any symbol key. Now press the same symbol key several times while you are still holding down C= .
5. Try the C= key and some of the other symbol keys.

Question:
Does holding down C= and pressing a symbol key cause the right or left symbol on the key to be printed?

Practice:

1. Try making different symbol patterns.
2. Can you add colors to your symbol patterns?
3. Restore the system to normal. Remember Section D, 5?

Computer License Test
Test 1

Introduction to the Keyboard

You must be able to do the following operations without using the manual:

1. Turn the computer and monitor on.
2. Clear the screen of any printing.
3. The teacher/tutor will type in 2 lines of printing (text). Be able to move the cursor up, down, to right, or to the left to make corrections.
4. Be able to *insert* a letter in text on the screen.
5. Be able to *delete* a letter in text on the screen.
6. Turn *reverse* printing on. Type your name. Turn reverse printing off.
7. Make bars of at least 3 different colors and then return the screen to normal.
8. Make at least 5 different graphic symbols and return the screen to normal.

Student name:_____

Test date_____

_____

_____

Date passed (100%)_____

Approved by_____

Part II:
My Computer Knows My Name.

An Introduction to Programming

1. If the "on" light is not red, turn the computer switch on
2. Turn on your monitor (T.V. screen)

A. Direct Statement

1. Type your name into the computer.
2. Press RETURN
   Did the computer print your name?
   The computer is telling you that it does not understand what you want it to do.
3. Type this—PRINT (your name goes here)
4. Press RETURN
   Was that any help?
5. Now type—PRINT "(your name)"
   (Remember to hold down the SHIFT key when you type the quotation mark")
6. Press RETURN
   Wasn't that better? What do you think made the difference?

Practice

Repeat the statement in line 5, using other names inside the quotation marks.

Special note: You need to know a special rule about using the cursor to correct an error if you make it inside quotation marks. It will not work the way it does when you do not have quotation marks.

Type—"TODAY IS THH FIRST (oops! you made a spelling error.)
Try using the cursor to go back and change THH to THE.

Stubborn, isn't it?
Now type the second quotation mark right there and then try moving the cursor back to make the change. It works! Now move the cursor back to the right, press the SPACE BAR over the quotation marks to erase them, and continue typing, DAY OF THE WEEK"

B. Simple Programs

Programs are the instructions you put into the computer's memory.

Type the following programs into the computer exactly as shown, *including* punctuation. We use punctuation marks to give special instructions to a computer. They do not mean the same as in our written language.

Remember: press RETURN at the end of each line
Now—locate the RUN/STOP key. Be ready to press it to interrupt the program. *Don't* forget to press it.

Note: GOTO is typed as *one word*

1. Type this:

NEW
10 PRINT "(your name)"
20 GOTO 10
RUN

(Don't forget to press the RUN/STOP key to stop the program)
Note: RUN is a *command* to tell the computer to do something with the instructions (program) you gave it

2. **NEW**
   10 PRINT "your name "; (Press the SPACE BAR once between your name and the quotation mark)
   20 GOTO 10
   RUN
   Notice the difference the semicolon made!
3. **NEW**
   10 PRINT "your name",
   20 GOTO 10
   RUN
   Now notice what the comma did!
4. Practice programs 1, 2, and 3 using different names.
5. Now here is an easier way to change line 10 without retyping the entire program.
   a. Type in any one of the first 3 programs, but *not* the command RUN
   b. Type 10 (the line number) PRINT (the instruction) " (to tell the computer to print what is enclosed) then use some *other* name besides your own. Remember to end with ". Press RETURN.
   c. Now type LIST and press RETURN
   See how the computer only remembered the last thing you wanted in line 10?
   d. Type RUN
6. Go back and practice program 5 several times.

C. Strings

A computer works with numbers, and when we want it to remember words we call them *strings*. We use the dollar sign $ to stand for the word *string*. We label these strings so the computer knows which one we are talking about. They can be called A$ or C$ or N$, whatever you want to call them.

1. Try this program. Don't forget the RUN/STOP key to end the program.
   NEW
   10 LET A$ = "your name"
   20 PRINT A$;
   30 GOTO 20
   RUN
2. Retype the program and substitute N$ or C$ or any other string label in both line 10 and line 20.

Question:

What happens if the two labels do not match? Why do you think this happened?
Remember to try different punctuation at the end of line 20.

D. INPUT Statement

INPUT means the computer is going to ask *you* to put something in by printing a ?. Try typing your name or some other word when it prints ? and then press RETURN.

1. Type in this program
   NEW
   10 INPUT A$
   20 PRINT A$;
   30 GOTO 20
   RUN
2. Note how changing the punctuation at the end of line 20 changes your screen print-out. Try some practice programs.

E. FOR-NEXT Loops

Here's how to tell the computer to SLOW DOWN!
We use a FOR-NEXT loop so it counts some empty lines between printing.

In this program the computer will count to 100 during the loop (lines 40 and 50)

1.
   NEW
   10 INPUT A$
   20 PRINT A$;
   30 FOR C = 1 TO 100
   40 NEXT C
   50 GOTO 20
   RUN
2. Type LIST
3. Retype line 30 using other numbers (instead of 100) in the FOR statement. Type LIST. Do you see the changed program? Now type RUN.

Questions:

What happens when you use numbers in the loop greater than 100?

What happens when you use numbers in the loop less than 100?

Computer License Test
Test 2

My Computer Knows My Name

Circle the correct answer to each question:

1. Which is the correct spelling for this computer command?
   GO TO **GOTO**

2. Strings are used when the computer works with
   quotation marks **numbers** **commas** **words**

3. Which command tells the computer to print out your whole program?
   RUN **LIST** **PRINT** **RETURN**

4. When you want the computer to ask *you* to type something in, which word do you type in the program?
   GOTO **RESET** **LIST** **INPUT**

5. Which command tells the computer to do what your program tells it to?
   GOTO **RUN** **PRINT** **NEW**

6. Each line in a program should have a
   string **line number** **input** **loop**

Here is a program:

```
NEW
10 PRINT "John Brown"
20 GOTO 10
RUN
```

Rewrite the program and put in a FOR-NEXT loop to slow down the program.

Student name:_____

Test date_____

_____

Date passed (100%)_____

Approved by_____

Using a computer-managed instructional software package written by the author and John Fiala, a student at Arkansas State University, the Jonesboro, Arkansas, public schools have added a new dimension to third-grade math.

MUPETing the PET for Third Grade Mathematics

by Thomas D. Bishop
Associate Professor of Mathematics
Arkansas State University
State University, Arkansas 72467

During the academic year 1981-82 the Jonesboro Public Schools, Jonesboro, Arkansas, received a Title IV-C grant to develop a model computer-assisted instructional management package. The funding was awarded through the Arkansas State Department of Education. The project was entitled C-AIM, an acronym for Computer-Assisted Instructional Management. A steering committee consisting of representatives from the Arkansas Department of Education, the Jonesboro Public Schools and Arkansas State University met several times to provide input into the program development. The decision was made to focus the project on third grade mathematics using the Arkansas Department of Education set of *Basic Skills Objectives* for mathematics K-8. The project development was organized into five major components. The components were inservice, selection of hardware, organization of pilot objectives, development of computer software and implementation.

INSERVICE

The first step in the project was to provide eighteen hours of inservice for every academic administrator in the school district. The district had Commodore 4016 microcomputers available in their instructional program in the high school, so these were used in the inservice program. The administrators were provided hands-on experiences with the microcomputers and the capabilities and limitations of the machines were explored. A presentation of characteristics of several different makes of machines and of different languages available was made by the inservice directors. Different applications of the microcomputer to the instructional program were demonstrated.

The teachers' inservice was organized into six sessions of three hours each and was conducted after the software for the computer management system had been written. The teachers will have the microcomputers available for use in their classrooms for computer-assisted instruction as well as for management.

Selection of Hardware

After the administrators' inservice, the process of selecting hardware was started. The hardware had to meet the minimum criterion of 16K bytes of random access memory. Other considerations were total memory capacity, expansion possibilities, disk operating capabilities, networking features, vendor support, sound, color, and compatibility of equipment with existing equipment within the district and region.

Objectives

C-AIM has been developed using the ADE *Basic Skills Objectives*. Since the third grade was selected as the target group, the mathematics objectives for grades 1-4 were organized into one set. This range was chosen due to the anticipated variance in third grade student mathematics skills. The objectives were placed in strands consistent with those identified by the state Department

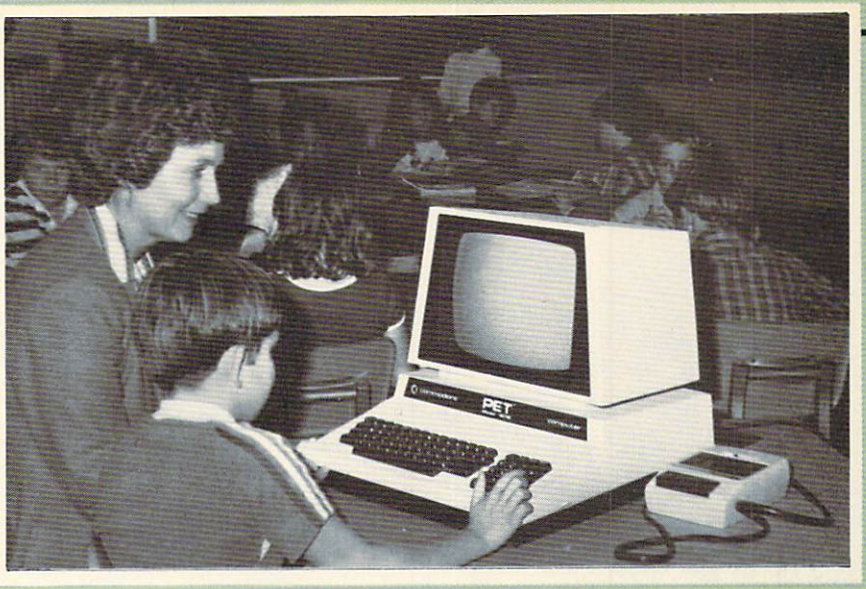
of Education and then the repetition within the set was eliminated as much as possible. This yielded a set of 149 different objectives. These objectives were then organized by function and placed in fairly homogeneous units. The units were then sequenced as much as possible. Each of the eighteen units has a testing program to cover the objectives of that unit.

Software Development

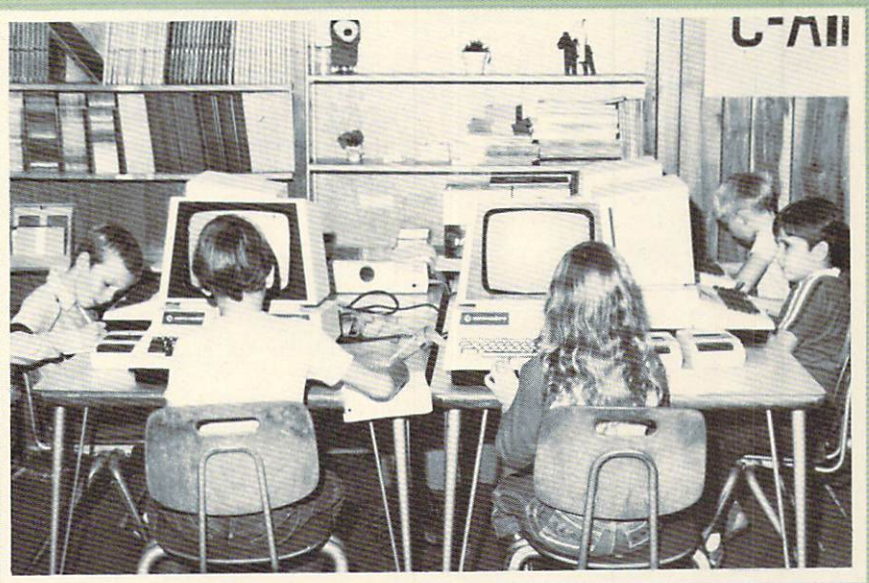
The C-AIM Software was written to have the system generate unit tests and then administer the tests via the microcomputer. C-AIM also maintains a database on each student that includes general information (name, age, etc.), standardized test score history, and current status on objectives and unit tests. The system requires a dual disk drive in order that testing programs may be stored on one diskette and the management system stored on the other. C-AIM allows for 949 student records on a single 5¼" diskette. Each record may include the base data as well as status on up to 427 individual objectives. C-AIM also includes processing branches for entering or deleting students and test management. The output branches allow for lists to be printed in a variety of selections in order to assist the teacher in organizing instruction. The output choices include a comprehensive review of an individual student record which may be used for parent-teacher conferences.

HARDWARE AND OPERATIONAL DESIGNS

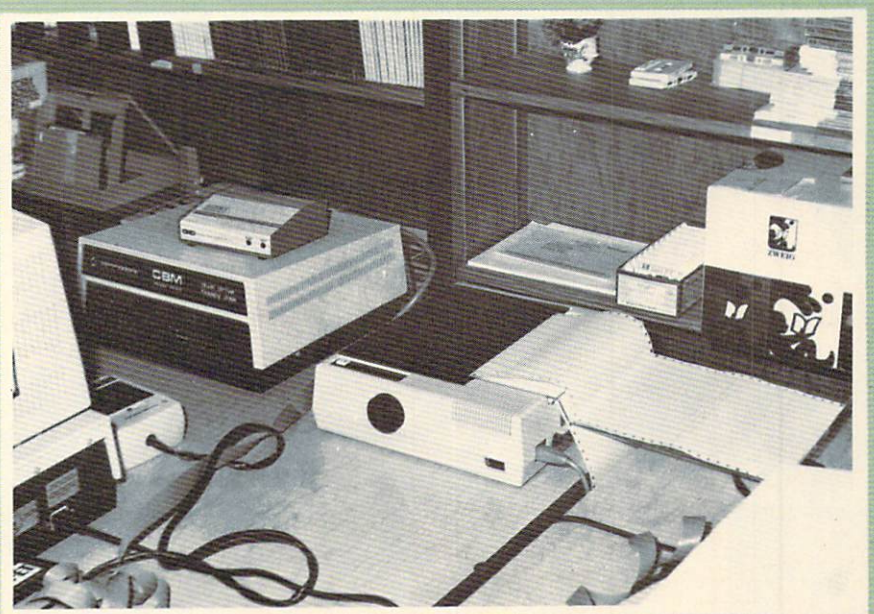
The hardware chosen for use with C-AIM was Commodore equipment. The primary considerations for this choice were the compatibility with existing equipment and the networking capabilities. The pilot being used for the C-AIM project allows for up to six students to be involved in testing at any one time.



Microcomputers may be pulled from network for classroom work.



Students testing in the "MUPET" lab.



C-AIM's Network.

Microcomputers

The C-AIM system utilizes from one to eight 4016 or 4032 microcomputers. C-AIM could be implemented using only one machine, but due to the number of students that must be tested in a typical situation, six machines is considered optimal. The microcomputers used in the C-AIM system have at least 16K of memory and level 4.0 BASIC.

MUPET

The MUPET is an IEEE multiplexer that allows up to eight Commodore microcomputers to access common IEEE peripherals (disk drives or printers). The MUPET requires no special commands or software for operation. The MUPET system used with C-AIM consists of the following components: (See Figure 1)

1 12 volt DC Power Supply
1 Double MUPET Module (allows access to a "common" disk drive and a "local" printer.)
5 MUPET modules (one on each microcomputer.)
6 IEEE cables six feet long

Peripheral Devices

Peripheral devices include:
1 Commodore CBM 4040 Dual Drive Floppy Disk
1 Commodore CBM 4022 Tractor Printer
2 PET to IEEE Cables

IMPLEMENTATION

C-AIM is being piloted during the 1982 school year at North Elementary School in Jonesboro Arkansas. All the third grade students are participating in the program and the reaction has been excellent. The C-AIM system is housed in a converted boiler room, affectionately referred to as the "MUPET Lab." This is where the testing and management functions take place. The microcomputers are also pulled from the system when not needed for testing or other group activities and are used for instruction in the individual classrooms. A photographer who entered the lab while students were testing commented that the students never noticed him while he was taking pictures because they were too intent on what was happening on the screen. The teachers' reaction has also been positive, although as expected, having such detailed information on students in a continuous fashion has markedly affected instructional planning. C=

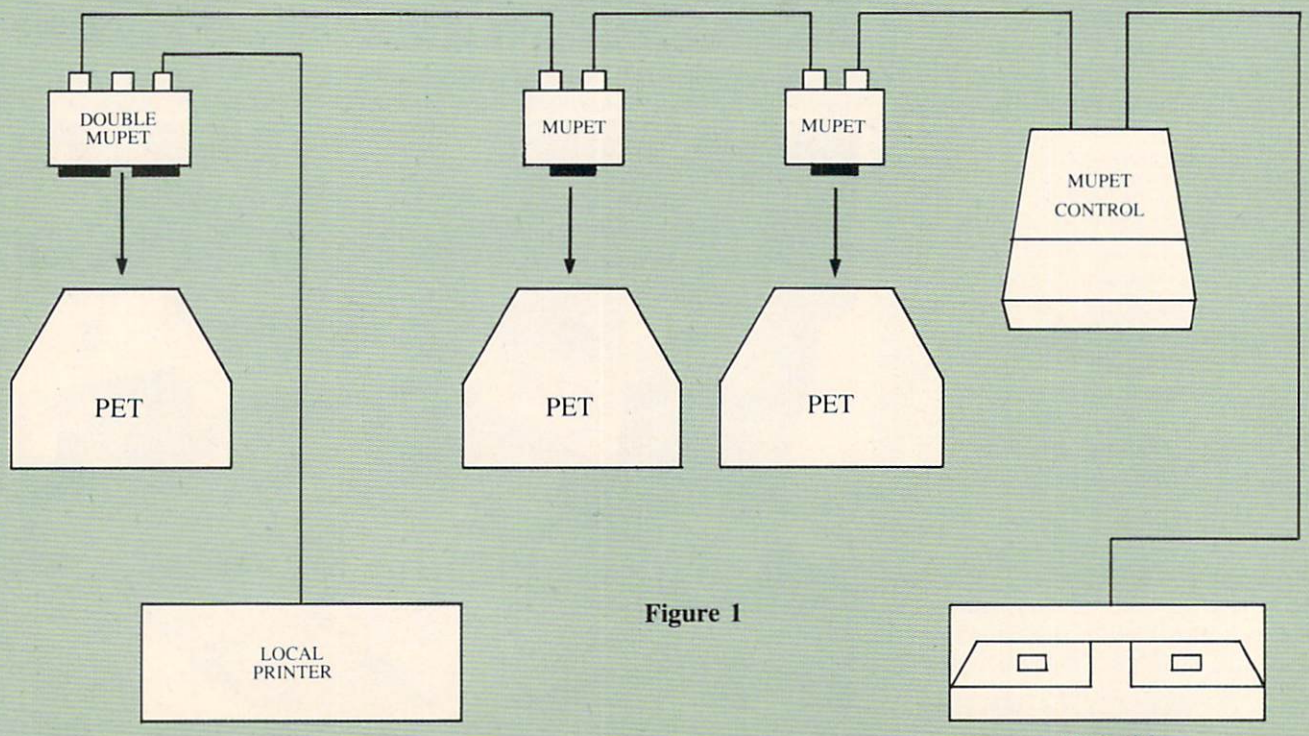


Figure 1

Converting PET Programs for the Commodore 64

BY GARRY KIZIAK
For the Canadian Commodore Educational Group

Many owners of the new Commodore 64 will have access to a large number of programs written originally for the PET computer. It is natural for these people to ask "What is involved in converting these programs so that they will run on the 64?". This article will attempt to detail some of the steps involved and hopefully make the conversion somewhat easier. I will only be discussing conversions involving 2.0 and 4.0 ROM PETs. Those interested in converting programs from 1.0 ROM PETs should be able to make the additional changes necessary.

In many cases, a PET program will run immediately on a 64. In some cases, a few minor changes will make the program workable. In a few cases, major surgery will be required, and in some instances, unless you are heavily into machine language, the conversion will be impossible. The type of conversion required will depend on the makeup of the original program.

As I said above, some programs will run immediately on the 64. These programs will be written entirely in BASIC and will not make use of the commands POKE, PEEK, WAIT, SYS, and USR. The easiest way to determine if a program falls in this category is to simply load the program into the 64 and run it. If it works, great. Otherwise read on.

Note: All BASIC programs for the PET will load into the 64 correctly. This may seem surprising since a PET program is stored in memory starting at location 1025 while 64 programs normally start at 2049. Such loads are successful because of a relocation feature incorporated into the Commodore 64 (and also the VIC 20) computer. These computers will automatically load a program at the START OF BASIC (wherever that happens to be), unless told to do otherwise (see your manual to see how to tell it to do otherwise).

I should point out that I have had some difficulty loading programs that were saved on a PET with 1.0 ROMS. Such programs do list but the first line is usually mangled. This can be fixed up by deleting that first line and retyping it or by a few simple POKEs.

The Simplest Conversion

Of the programs that do require conversion, the simplest to fix are the ones that do not use the SYS or USR commands. They may use the POKE, PEEK or WAIT commands, but these can usually be fixed up by changing an appropriate address and possibly a corresponding numeric value.

For example, POKE 59468,14 is a command frequently found in PET programs to convert the screen display to lower case. If this command is executed on the 64, nothing drastic will happen but lower case is definitely not displayed. The correct command on the 64 is POKE 53272,23. Thus part of the conversion process will be to locate all POKE 59468,14 statements in the program and change them to POKE 53272,23. Similarly, all POKE 59468,12 statements will have to be changed to POKE 53272,21 (This converts the screen display to upper case and graphics).

The majority of 'fixes' can be achieved in this manner.

i) Find the address on the PET that is causing a problem.
ii) Find the corresponding address on the 64.
& iii) Make all changes involving that address.

What is needed then is a list of addresses for the PET that can cause problems and a list of the corresponding addresses for the 64.

Actually, with a little more work, we can even do better. Ideally, a program should be able to run on any machine—PET with 2.0 ROMS, PET with 4.0 ROMS, and the Commodore 64.

This can be achieved for the upper/lower case conversion above in the following way.

Assume first that the program is running on a PET. Somehow have the computer execute the following commands:

```
3000 TEXT = 59468:  REM Address to be poked for
                        upper/lower case
3010 UC = 12:       REM Value to be poked for
                        upper case
3020 LC = 14:       REM Value to be poked for
                        lower case
```

On the other hand, if the program is running on a 64, have it execute the following:

```
3100 TEXT = 53272
3110 UC = 21
3120 LC = 23
```

Now change all POKE 59468,12 statements to POKE TEXT,UC and all POKE 59468,14 statements to POKE TEXT,LC. After these changes are made, the correct case will be displayed regardless of which computer the program is running on. If all other problem addresses can be fixed up in this manner, then we are well on our way to converting the program to work on all three computers.

Which Computer are You?

The first task then is to somehow identify what type of computer a program is running on.

There already is a standard technique for identifying whether a PET has 2.0 ROMS or 4.0 ROMS; namely,

```
110 IF PEEK (50003) = 160 THEN ... : REM 4.0
    ROMS
120 IF PEEK (50003) = 1 THEN ... : REM 2.0
    ROMS
```

PEEKing location 50003 on a 64 will usually yield a zero. I say 'usually' because 50003 is a RAM location on the 64 and is normally unused. However, machine language routines can be placed in that area and so you cannot be 100% sure what location 50003 will contain. The sequence below will set around this problem and will identify the type of computer correctly without destroying any machine code already there.

```
100 X = PEEK (50003): POKE 50003,0:Y = PEEK
    (50003)
110 IF Y = 160 THEN COMP$ = "4.0":REM 4.0
    ROMS
120 IF Y = 1 THEN COMP$ = "2.0":REM 2.0 ROMS
130 IF Y = 0 THEN POKE 50003,X:COMP$ = "64":
    REM COMMODORE 64
```

The statement POKE 50003,0 in line 100 has absolutely no effect on 2.0 PETS or 4.0 PETS since location 50003 is in ROM. On the 64 however, it puts a zero into that RAM location. Notice that the original value in location 50003 is saved by the statement X = PEEK(50003) and restored again in line 130 if the computer is identified as being a 64. Note the use of the variable COMP$ to identify the type of computer just in case it is needed again later in the program.

Now the conversion process should be clear. It should include the following:

1) At the beginning of the program, jump to a subroutine that identifies the type of computer that the program is currently running on.
2) In that subroutine, initialize a set of standard variables (such as TEXT, LC, UC, etc.) to the correct values for that computer.
3) Change all references to numerical addresses or values to the corresponding standard variables.

Here is a sample initialization routine.

```
10 GOSUB 60000
20 REM MAIN PROGRAM

60000 X = PEEK(50003):POKE 50003,0:Y =
    PEEK(50003)
60010 REM INITIALIZE VARIABLES COMMON
    TO 2.0 & 4.0 PETS
60020 TEXT = 59468:UC = 12:LC = 14:SCREEN =
    32768:HIV = 144
60030 NUMCHAR = 158:KEY = 151:NOKEY = 255
60040 IF Y < > 1 THEN 60100
60050 REM INITIALIZE VARIABLES PECULIAR
    TO 2.0 PETS
60060 COMP$ = "2.0":ENA = 46:DIS = 49
60070 RETURN
60100 IF Y < > 160 THEN 60200
60110 REM INITIALIZE VARIABLES PECULIAR
    TO 4.0 PETS
60120 COMP$ = "4.0":ENA = 85:DIS = 88
60130 RETURN
60200 IF Y < > 0 THEN 60300
60210 REM INITIALIZE VARIABLES PECULIAR
```

```
TO THE 64
60220 COMP$ = "64":TEXT = 53272:UC = 21:LC =
  23:SCREEN = 1024:HIV = 788
60230 NUMCHAR = 198:KEY = 203:NOKEY =
  64:ENA = 49:DIS = 52
60240 POKE 50003,X:RETURN
60300 PRINT "I DON'T RECOGNIZE THIS COM-
  PUTER.":END
```

The variables SCREEN, NUMCHAR, etc. will be explained shortly.

More Problem Areas

Upper/lower case conversion is certainly not the only problem area. Another potential one is the screen.

1. The Screen

On the PET the screen is found in memory locations 32768-33767. On the 64, it is found in locations 1024-2023.

If all output to the screen is obtained through the use of PRINT statements, then absolutely no problem will arise. If however, the output is POKEd to the screen, then changes will be required.

These changes are best achieved by assigning a value to the base address of the screen and then using an appropriate offset from that base.

For example the base address of the screen on the PET is 32768 while on the 64 it is 1024. Therefore, the first thing to do is to assign values to the standard variable SCREEN as follows:

SCREEN = 32768 if on a PET
SCREEN = 1024 if on a COMMODORE 64

i) Poking a single value onto the screen.

A statement of the form

POKE 32956,61

on a PET has to be changed as follows:

First, calculate the offset.

Offset = 32956–32768
= 188

Then, change POKE 32956,61 to
POKE SCREEN + 188,61

The resulting statement will work on either a PET or a 64 (assuming SCREEN has been properly initialized).

Notice that the 61 does not have to be changed as these values are the same for both PETS and the 64

ii) Poking with a loop.

The following is a typical PET routine that POKES a border of 'reversed diamonds' around the screen.

```
100 FOR I = 32768 TO 32807 : POKE I, 218 : NEXT
110 FOR I = 32847 TO 33767 STEP 40 : POKE I,218 :
  NEXT
120 FOR I = 33766 TO 33328 STEP – 1 : POKE I,218
  : NEXT
130 FOR I = 33688 TO 32768 STEP – 40 : POKE
  I,218 : NEXT
```

This can be changed to work on both PETS and 64 by changing each screen address as above.

```
100 FOR I = SCREEN TO SCREEN + 39 : POKE
  I,218 : NEXT
110 FOR I = SCREEN + 79 TO SCREEN + 999 STEP
  40 : POKE I,218 : NEXT
120 FOR I = SCREEN + 998 TO SCREEN + 990 STEP
  – 1 : POKE I,218 : NEXT
130 FOR I = SCREEN + 920 TO SCREEN STEP – 40
  : POKE I,218 : NEXT
```

Or better yet

```
100 FOR I = 0 TO 39 : POKE SCREEN + I,218 :
  NEXT
110 FOR I = 1 TO 24 : POKE SCREEN + 39 +
  I*40,218 : NEXT
120 FOR I = 38 TO 0 STEP – 1 : POKE
  SCREEN + 960 + I,218 : NEXT
130 FOR I = 23 TO 1 STEP – 1 : POKE SCREEN +
  I*40,218 : NEXT
```

2. Clearing the Keyboard Buffer

The PET is able to retain up to ten keystrokes in a buffer, enabling you touch typists to type as fast as you can without losing any keystrokes. This can sometimes add extra unwanted characters to the beginning of an input, so a common technique in PET programming is to clear the keyboard buffer before each input is requested. This can be accomplished in a couple of ways.

```
100 FOR I = 1 TO 10 : GET A$ : NEXT
```

or

```
100 POKE 158,0
```

The first method will work as is on the 64. The second method must be changed.

On 2.0 and 4.0 PETS, location 158 always contains the number of characters in the keyboard buffer. On the 64 this value is stored in location 198. Thus if we assign values to the standard variable NUMCHAR as follows:

NUMCHAR = 158 if on a PET
NUMCHAR = 198 if on a 64

and change all references to POKE 158,0 to POKE NUMCHAR,0, then the resulting statement will work on both computers.

3. Pausing Until any Key is Pressed

Here again two techniques are commonly used.

```
100 GET A$ : IF A$ = "" THEN 100
```

is certainly the simplest and will work on both computers.

100 POKE 158,0 : WAIT 158,1 : POKE 158,0

is another technique and will have to be changed to

100 POKE NUMCHAR,0 : WAIT NUMCHAR,1 : POKE NUMCHAR,0

4. Which Key is Pressed

A common technique used on the PET, especially in games, is to PEEK at location 151 to see if a key is being pressed and if so which one. Depending on which key is pressed a certain action is performed. This technique is frequently used in games that use the numeric keypad as a joystick. A sample sequence might be

```
500 X = PEEK(151)
510 IF X = 255 THEN 1000 : REM NO KEYPRESS
520 IF X = 18 THEN 2000 : REM 2 KEY IS PRESSED
530 IF X = 50 THEN 3000 : REM 8 KEY IS PRESSED
  etc.
```

The conversion here is a little more complicated but is still possible. First, we need to know that location 151 on the PET corresponds to location 203 on the 64. Then assign the following values to the standard variable KEY:

KEY = 151 if on a PET
KEY = 203 if on a 64

Replacing line 500 with

500 X = PEEK(KEY)

gives us a start with the conversion.

Another problem occurs with the values stored in location 151 (or 203) when a key is not being pressed. Location 151 on the PET contains 255 while location 203 on the 64 contains 64. This time we will use the standard variable NOKEY and initialize it as follows:

NOKEY = 255 if on a PET
NOKEY = 64 if on a 64

Line 510 is then replaced with

510 IF X = NOKEY THEN 1000

There are two problems associated with the other keys. First, location 151 will contain a certain value on the 2.0 machines, the same value on the Skinny 40 (i.e. the 9 inch screen) machines, but a different value on the Fat 40 machines. There is no standard way, that I am aware of, for distinguishing between a Skinny 40 and a Fat 40 machine. But PEEKing at location 57344 will do as well as any other. On a Skinny 40 you will get a value of 169 while on a Fat 40 you will get a value of 76.

The easiest way to see what value is stored in location 151 is to run the following program segment and press any key that you wish to test.

```
100 KEY = 151 : REM KEY = 203 ON THE 64
110 PRINT PEEK(KEY) : GOTO 110
```

The second problem arises from the fact that the 64 does not have a numeric keypad and using the numbers 2,4,6, and 8 to simulate a joystick is unacceptable. It would be much better to use the keys I,J,K, and M or some other suitably arranged set of keys.

My suggestion for getting around this is to first settle on the keys that you wish to use on each machine (they don't have to be the same). Find the values corresponding to these keys by running the short program above and store these values in corresponding standard variables which I like to designate K1, K2, K3, etc. (for KEY1, KEY2, KEY3, etc.) Then lines 520 and 530 can be replaced by the following:

```
520 IF X = K1 THEN 2000
530 IF X = K2 THEN 3000
  etc.
```

5. Disabling the Stop Key

The stop key on the PET can be disabled by altering the Hardware Interrupt Vector. For example.

POKE 144,49 for 2.0 PETS
POKE 144,88 for 4.0 PETS

will disable the stop key (and the time clock as well).

The corresponding command on the 64 is

POKE 788,52

To enable the stop key again

POKE 144,46 for 2.0 PETS
POKE 144,85 for 4.0 PETS
and POKE 788,49 for the 64

These can be replaced by

POKE HIV,DIS to disable the stop key
and POKE HIV,ENA to enable the stop key

after appropriately initializing the variables HIV, DIS, and ENA. On the 64, the program can still be stopped by pressing the RUN/STOP and RESTORE keys simultaneously, but this will prevent stoppage of a program due to accidently pressing the STOP key.

A good question to ask is "How do you know what value is to be stored in these locations?". The PET program actually tells you the location to POKE as well as the value, but the value to be POKED on the 64 is usually different (c.f. disabling the stop key above or converting to upper/lower case). A memory map will tell you what location to POKE on the 64, but it will not tell you what value to POKE it with. A good start is to PEEK that location from direct mode and make note of the value. Do this for all three machines and it will tell you the 'normal' state of that location. For example, PEEKing loca-

tion 144 on 2.0 PETS and 4.0 PETS yields 46 and 85 respectively. PEEKing at 788 on the 64 yields 49. Observing that the value to disable the stop key on the 2.0 & 4.0 PETS are each three more than the 'normal' value, a good start to finding the correct value on the 64 is to add 3 to the 'normal' value of 49, obtaining 52. This process will work for more than 90% of the problem values. It is that last 5–10% that makes the conversion challenging.

It would be impossible to list all problem locations and their 'fixes' here (I will list what I feel are the more common ones below). Instead I have attempted to give you a feeling for how the conversion should proceed. The proper tools that are required are the excellent memory maps (both zero page and ROM routines) published for all three computers in COMPUTE! by Jim Butterfield. Another excellent source is the book *Programming the PET/CBM* by Raeto Collin West and I am sure there are others.

Some of the More Common Problem Locations

2.0 PET	Location on 4.0 PET	64	Suggested Name	Description
40–41	40–41	43–44	SBAS	Start of BASIC text
42–43	42–43	45–46	SVAR	Start of variables
44–45	44–45	47–48	SARR	Start of arrays
46–47	46–47	49–50	EARR	End of arrays
52–53	52–53	55–56	TMEM	Top of memory
144–145	144–145	788–789	HIV	Hardware Interrupt Vector
151	151	203	KEY	Which key is pressed
158	158	198	NUMCHAR	Number of characters in keyboard buffer
159	159	199	RVS	Screen reverse flag
167	167	204	CRSR	Flag for flashing cursor in GET statements
196	196	209	SLO	Pointer to screen line (low/high format)
197	197	210	SHI	
198	198	211	CH	Horizontal position of cursor
216	216	214	CV	Vertical position of curosr
623	623	631	BUFF	Start of keyboard buffer
634	634	—	—	Start of first cassette buffer
826	826	828	CAS	Start of second cassette buffer
32768	32768	1024	SCREEN	Start of screen memory
59468	59468	53272	TEXT	Poke location for upper/lower case
64721	64790	64738	—	Simulates power on reset
			Special Values	
12	12	21	UC	Upper case
14	14	23	LC	Lower case
255	255	64	NOKEY	No key is pressed
46	85	49	ENA	Enable stop key
49	88	52	DIS	Disable stop key

A Couple of Cautions

1. The PET and the 64 only recognize the first two letters of a variable name. When converting a program you must make certain that the variables already present in the program do not conflict with the standard variables suggested above. If there is a conflict, change whichever you feel is easier.
2. If a program is to be used both on a PET and a 64, then the changes should be made on and saved with a PET computer. The reason for this is that a program saved on a 64 will not load properly on a PET, due to the lack of a relocation feature in the PET (But see below).

   If the changes are made on a PET, then a utility such as BASIC AID or POWER will be invaluable since you can type in such things as

   **FIND \ POKE **

   and all lines that contain a POKE statement will be listed, making it easier for you to make the necessary changes and to make certain that you have found all of them.

   Similarly you can type in

   **FIND \ SC **

   to see if there are any variables in the program that will conflict with the standard variable SCREEN.

Programs That Contain SYS or USR Commands

These programs will require that you be somewhat familiar with machine language in order for you to be able to make the necessary conversions. Such changes are beyond the scope of this article. However, let me say that these M/L routines themselves fall into a number of categories.

1) The routine works on the 64 as is. (Few routines will likely fall in this category)
2) The routine will work with a simple address change. (These are frequently ROM routines such as the routine for resetting the entire stack)
3) The routine will work with some minor changes. An example here could be a routine to reverse a portion of the screen. Chances are the only changes necessary would be for the location that determines the base address of the screen. However, if parameters are passed in the calling statement, then the location of certain ROM routines (such as checking for a comma) might have to be changed as well.
4) The routine will require major surgery before it will work. A program like BASIC AID or MICROMON would fall into this category. Such programs should be left to the experienced users.

Other areas that may require major changes are those programs that make use of the BASIC 4.0 disk commands. Some of these can be fixed up easily, but some are extremely difficult (e.g., those that make use of Relative Record files).

Programs that make use of CB2 sound will still run on the 64, but no sound will be produced. The POKEs that the PET uses to produce these sounds will POKE into the ROMS of the 64 and hence do no harm. Once you are familiar with the sound process on the 64, here is a good place to make use of the variable COMP$. For example, suppose lines 1000–1030 in the PET program are used to produce the sound. Leave these lines exactly as they are and add a similar routine for producing sound on the 64 beginning with

1031 IF COMP$ < > "64" THEN 1040

Your sound routine can be placed in lines 1032–1039 and the rest of the program should proceed as normal.

Loading Programs Saved on the 64 Into the PET

As mentioned above, programs saved on the 64 do not load properly into a PET. It is not a difficult procedure to correct this shortcoming however. Here are the steps.

1. Type in a dummy line 0 into the PET.
   0 REM
   will do.
2. Type in **POKE 2048,0**
3. LOAD in the program that was saved on the 64 as you normally would.
4. Type in **POKE 1025,1 : POKE 1026,8**

You should now be able to LIST the program including the dummy line 0 that you typed in initially. Delete this line by typing in 0 <RETURN>. The process is now complete. You can save the program to cassette or disk. The next time that you load it into your PET, it will load normally. If you are using a disk, you will notice that the program is 3 blocks longer than the original even though it is the same program. The reason for this is that the Start of Variables pointer did not get changed properly. An experienced programmer can set into the monitor and make the necessary changes without too much difficulty, but the program will operate correctly without making this change. C=

WHAT'S SMALLER THAN A BUSINESS CARD?

FASTER THAN CASSETTES?

AND FAR LESS EXPENSIVE THAN DISKS?

Why the ESF-20/64 Stringy Floppy from Exatron, of course. Our exciting little storage alternative gives you near floppy disk speed and reliability at a budget-minded price. Our high quality digital mass storage system is the perfect product to fill the gap between cassettes and floppy disks. And that's especially true in the case of the Commodore VIC-20® and Commodore 64® microcomputers.

The Exatron Stringy Floppy system is based on a small endless loop tape cartridge we call "The Wafer". This wafer measures only 2 3/4" x 1" x 3/16" — or about the dimensions of a standard business card. Wafer was born to run fast — at around 7200 baud, or 14 times faster than a standard cassette and has a memory capacity of 64K bytes. The ESF-20/64 system costs less than $200 and wafers are less than $3.00 each. No wonder computer people call us the "poor man's disk system".

If you'd like to get the world on a string, and bring your VIC-20® or Commodore 64® into the 20th Century, just fill out the coupon below and mail it to EXATRON, 181 Commercial Street, Sunnyvale, CA 94086. If you need to place an order call (408) 737-7111, outside California (800) 538-8559 and ask for Chrissy or Natalie.

The ESF-20/64 Stringy Floppy System from Exatron for use with the Commodore VIC-20® and Commodore 64® microcomputer.

exatron

For exciting alternatives in mass storage.

Name _____

Address _____

City/State/Zip _____

Mail today for information on the ESF-20/64 from EXATRON, 181 Commercial St., Sunnyvale, CA 94086

The VIC Magician
Peeking Keys To Control Actions

By Michael S. Tomczyk
Product Marketing Manager

Here we are again . . . with more lessons to help you write your own games. As we've said before, many of the best gamewriting techniques aren't explained anywhere.

In this article we're going to explore several aspects of gamewriting that are hard to pick up if you're just getting started on your own—starting with a summary of how to make symbols appear AT RANDOM in various places on the screen, how to make them MOVE, and finally, how to "PEEK" ahead to establish "collision detection."

It will help a lot if you have your *VIC-20 Programmer's Reference Guide* or VIC owner's guide handy . . . and if you missed Part I and Part II in this gamewriting series, take a look at the review material on page 52 before going on.

Making Objects Appear At Random On The Screen

In Part I and II, we touched on how to make objects appear at random but now it's time to summarize that information. It never hurts to review how random numbers work, and the chart below will give you a quick reference when you want to use this "random" technique in your action games.

Most action games use the VIC's built-in random number generator to make objects appear haphazardly along one of the four screen borders (top, bottom, left and right), and then move across the screen. You can also make objects appear at random anywhere on the ENTIRE SCREEN, or in a particular row or column.

So you can better see what we're doing, let's use the variable TP to stand for the TOP ROW LOCATION on the screen, BT for the BOTTOM ROW LOCATION, LF for the LEFT COLUMN LOCATION, RT for the RIGHT COLUMN LOCATION and A for ANYWHERE ON THE ENTIRE SCREEN. **It's important to remember that we're talking about SCREEN LOCATION of a symbol. Study the following chart before reading on:**

FORMULA	HOW TO USE IN A GAME
TP=(22*RND(1))+7680	**TOP ROW:** Using TP as a screen location puts a symbol at a random position on the TOP ROW of the screen.
BT=(22*RND(1))+8164	**BOTTOM ROW:** Using BT as a screen location puts a symbol at a random position on the BOTTOM ROW of the screen.
L=INT(22*RND(1))+0 LF=7680—(22*L)	**LEFT COLUMN:** Using LF as a screen location puts a symbol at a random position along the LEFT SIDE of the screen. The first "L" formula picks a number from 1 to 23 which is needed in the LF formula.
R=INT(22*RND(1))+0 RT=7701—(22*R)	**RIGHT COLUMN:** Using RT as a screen location puts a symbol at a random position along the RIGHT SIDE of the screen. The first "R" formula picks a number from 1 to 23 which is needed in the RT formula.
A=(506*RND(1))+7680	**ANYWHERE ON THE SCREEN:** Using A as a screen location puts a symbol at a random position anywhere on the entire screen (note there are 506 locations on the VIC screen).

Let's see how these formulas work in a BASIC program. Type the word NEW and press return to erase any previous programs that might be in your computer, then enter this program exactly as shown:

```
10 PRINT" SHIFT CLR/HOME RANDOM SYMBOLS"
     (press RETURN)
20 TP=(22*RND(1))+7680
     (press RETURN)
30 BT=(22*RND(1))+8164
     (press RETURN)
40 L=INT(22*RND(1))+0:LF=7680+(22*L)
     (press RETURN)
50 R=INT(22*RND(1))+0:RT=7701+(22*R)
     (press RETURN)
60 A=(506*RND(1))+7680
     (press RETURN)
70 TC=TP+30720:POKETP,90:POKETC,4:GOTO20
     (press RETURN)
```

Whew! Lots of typing, huh? Well, we'll do a lot of things with this program so you won't have to keep retyping all those tricky formulas. (The VIC MAGICIAN doesn't like retyping formulas any more than you do!)

Type the word RUN and press the RETURN key to see the program work.

What happens? A series of red diamonds are appearing along the top of your screen! That's because in Line 70, we used the TOP ROW FORMULA we set up in Line 20 to put a random series of symbols on the screen. We got the POKE NUMBER (90) for the diamond symbol from the SCREEN CODE chart on Page 268 of your *Programmer's Reference Guide*, page 142 of your VIC owner's guide. Press the RUN/STOP key to stop your program and type LIST and hit RETURN to see the listing. Here's a line-by-line explanation of the program:

LINE 10 clears the screen and displays a title.

LINE 20 sets up a random formula for the TOP ROW screen location, which we've designated TP. You can use your own variables instead of these, incidentally (for instance, T, T1, TR or any other variable works just fine).

LINE 30 sets up a random formula for the BOTTOM ROW.

LINE 40 sets up a random formula for the LEFT COLUMN.

LINE 50 sets up a random formula for the RIGHT COLUMN.

LINE 60 sets up a random formula for ANYWHERE ON THE SCREEN.

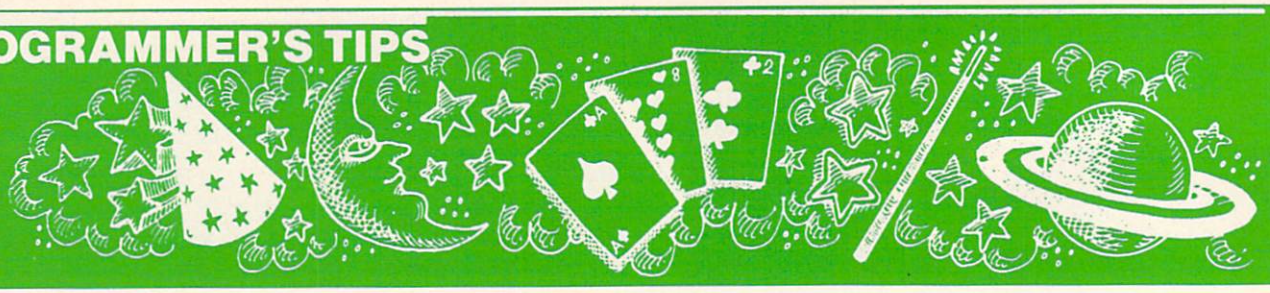
LINE 70 starts out by establishing the COLOR LOCATION which matches the TOP ROW LOCATION. The color location of any symbol we are POKEing on the screen is always the number 30720 added to the screen location. So the color location of location TP must be TP + 30720. We will call the color location TC (for "TOP ROW COLOR") and get back to this in a moment.

Remember, you're using TP as a SCREEN LOCATION for the symbol you want to display AT RANDOM in the TOP ROW. We used the diamond symbol whose POKE NUMBER is 90 (from the SCREEN CODE chart in your manual), so we POKE TP,90 to display the symbol on the screen . . . but wait! To display a symbol you always have to make TWO POKES . . . the screen location AND the matching color location. So we'll POKE TC (the name we gave the color location for the top row) with the COLOR NUMBER 4 which is red (color numbers are always one less than the numbers shown on the keyboard, when POKEing colors). Finally, we GOTO Line 20 which sends the program back to get a NEW RANDOM NUMBER for the top row . . . and by the time the program comes back to Line 70, the variable TP has become a new random number . . . so the color location (TP+30720) also changes . . . and everything keeps going.

Confused? If so, take a look at the review on page 52 for more information on how Line 70 works. Once again . . . we defined the color location by adding 30720 to the screen location, which in this program is a RANDOM location called TP. We then POKE the screen location (TP) with the screen code number of the SYMBOL we want to display (in this case, a diamond whose screen code number is 90). Then we POKE the matching color location (TC) with a COLOR NUMBER, which is number 4 which is red. Try changing the color and symbol numbers (4 and 90) to get used to working with various settings.

Okay, you've seen the top row . . . how about the others? Well, all the formulas are still there in your program, although we only used one of them for this example. To use any of the other formulas, simply change Line 70.

To change a line, all you have to do is retype the entire line, including the line number, and press the RETURN key. So . . . retype Line 70 as follows to see the BOTTOM ROW example:



```
70 BC=BT+30720:POKEBT, 90:POKEBC,4:GOTO20
   (Press RETURN)
```

Type RUN and press RETURN. Notice you can still GOTO 20 in Line 70, because when the program goes there, it will "fall through" to Line 30 where the RANDOM BOTTOM ROW LOCATION is being set. Press the RUN/STOP key, then retype Line 70 as follows:

```
70 LC=LF+30720:POKELF, 90:POKELC,4:GOTO20
   (Press RETURN)
```

Type RUN and press RETURN. To try the next example, press the RUN/STOP key and retype Line 70 again as shown, then type RUN to see it work:

```
70 RC=RT+30720:POKERT, 90:POKERC,4:GOTO20
   (Press RETURN)
```

And . . . to see the "anywhere" example, try this Line 70:

```
70 AC=A+30720:POKEA,90:POKEAC,4:GOTO20
```

Adding a Bit of Music

You can add music to the examples we've just explored. Here's how: just add the "music lines" at Lines 80 and 85 and move the "GOTO" at the end of Line 70 to Line 90, as shown:

```
70 AC=A+30720:POKEA,90:POKEAC,4
   (press RETURN)
80 POKE36878,15:POKE36876,
   (100*RND(1))+150
85 FORT=1TO200:NEXT:POKE36876,0
90 GOTO20
```

Careful! This one will drive you bonkers if you listen to it long enough! In any event, here's how you did it:

LINE 70 is the same "ANYWHERE" example we used last. You simply deleted the GOTO 20 from the end of the line.

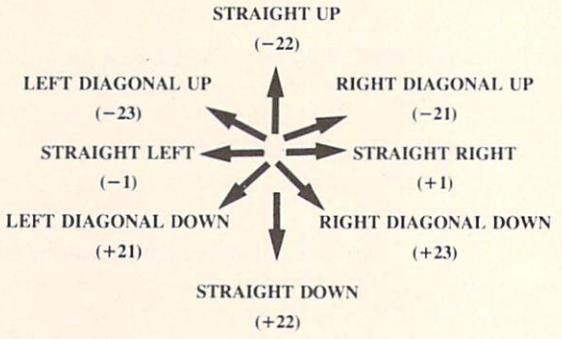
LINE 80 was automatically added to your program when you typed it in. POKE36878,15 sets the VOLUME at its highest level. POKE36875 is the POKE for "speaker 2" of the VIC's four internal "speakers." Usually, you POKE a speaker setting with a number from 128 to 255 to generate a musical tone. Here, instead of using one of those numbers, we used a RANDOM NUMBER to represent the musical tone. The RANGE of random musical settings is from 150 to 250. The 150 is the STARTING POINT and the 100 is the RANGE (150+100 = 250). The VIC will choose a different number from 150 to 250 each time it goes around and selects another diamond and plays the tone. The FOR...NEXT loop provides the DURATION the note plays, before we turn it OFF by POKEing36876,0.

LINE 90 is where we put the GOTO. We basically inserted the musical note being played in Line 80 after Line 70, but before the GOTO because we want to play the note before we go back around and display another diamond (and play another random note).

Are you beginning to get comfortable with random numbers? Just remember that the FIRST NUMBER on the left side of the formula is always the RANGE and the last number on the far right side of the formula is the STARTING POINT. So, for example, (506*RND(1))+7680 means we are selecting a random number starting at number 7680 with a range of 506, which means that the random numbers will be chosen from the set of numbers which includes 7680 to 7680+506, or 7680 to 8186.

Moving Symbols

To move a symbol on the screen, simply change the SCREEN LOCATION (the color location should automatically change because it is always the screen location plus 30720). Here, briefly, is how much you have to add or subtract to a screen location to move a symbol in one of the 8 possible directions:



Often, you may have a symbol appear at random on the left edge of the screen, and "fly" across the screen to the right. This is an easy movement because you simply add 1 to the screen location. You can add 1 to the location even if the location is "randomly" generated. Let's try an example:

We're going to retype our sample program a little, starting with a new Line 65. .STOP and LIST your program, then retype Lines 65-90 as shown:

```
65 E=LF+22
70 LC=LF+30720:POKELF−1,32:POKELF,90
   :POKELC,4
80 POKE36878,15:POKE36876,160:FORT=
   1TO10:NEXT:POKE36876,0
```

```
90 LF=LF+1:LC=LC+1:IFLF=ETHENGOTO20
100 GOTO70
```

Type the word RUN and press RETURN to see the program work, then press the RUN/STOP key, type the word LIST and press RETURN.

Look complicated? It's not! Did you ever read a *Commodore* magazine and get boggled trying to understand all the variables like LF, RT, A, LC, etc.?? Pretty confusing, huh? You may not have realized it yet, but you have just written one of those "complicated" programs yourself! And you understand what all the variables mean (at least you can check back to the chart if you forget any). Well, now we're going to start USING some of those variables to make your diamond symbol move, and do some other interesting things as well. First let's look at these extra program lines:

LINE 65 is a special line which defines a NEW SCREEN POSITION CALLED "E" (for "END" of the line). E at this point in the program becomes the SCREEN LOCATION AT THE END OF THE LINE where the random LF begins. For example, if LF starts out at position 8010, then E is defined as 8010+22. or 8032 which is the beginning of the next line. In a little while, we'll use E as an "end of line limiter."

LINE 70 is the same as the LEFT COLUMN example we used earlier, with the exception that we added: POKELF-1,32. This ERASES the PREVIOUS SYMBOL. In other words, we are POKEing one symbol in the first (left) column, then POKEing the next symbol in the next position (LF+1). If we don't erase the previous symbol, we will wind up with a line of symbols stretching across the screen and that's not what we want because we're trying to create the illusion of movement, which is what animation really is. So we must ERASE LF−1 by POKEing a SPACE SYMBOL (number 32) into the previous location. That's what this POKE does.

LINE 80 is our music program except we've speeded it up by changing the 200 in the FOR...NEXT loop to 10. You can speed up or slow down the rate at which the diamonds move (and the duration of the musical tone) by changing this number. We also changed the tone from a random tone to a steady tone (the number 160 is the number of the note we're playing here).

LINE 90 is a little tricky, so follow closely. The first thing we did was use our DIRECTION FORMULA which is shown in the "arrows" diagram above. You can see from the diagram that if you want a symbol to move STRAIGHT RIGHT across the screen, you have to increase the screen location by adding 1. That's what LF=LF+1 means. Of course, to keep the color location matching you have to increase that by one, also. We also set an END OF LINE LIMIT by using an IF...THEN statement. This statement translates, "If the screen position of the diamond (LF) reaches the end of the line and tries to wrap around to the first location on the next line (E) then GOTO Line 20 and choose a new random location. Note that the diamond stays in the last position on the line because we only erased LF minus 1, not LF. The last position is still LF so the diamond stays there. Is this beginning to resemble a game?

Line 100 sends the program back to Line 70 to POKE the NEW LF (caused by LF=LF+1 in Line 90). There are two GOTOs involved here. GOTO70 sends the program back to move LF one space to the right . . . and GOTO20 in Line 90 sends the program back to get a new random starting point if the diamonds reach the end of the line.

That should be enough to give you something to experiment with, but let's add a few more intriguing ideas before leaving you. You probably noticed that sometimes the little diamonds repeat themselves on lines where there is already a diamond. That's because everything is happening COMPLETELY at random. You can make the diamonds avoid lines which already have diamonds on them by adding this line to your program:

```
95 IFPEEK(E)=90THENGOTO20
```

The PEEK command may be new to you . . . it's explained in the accompanying VIC Magician article in this issue . . . and in this program we are "PEEKing" at the E-1 location (the last position on the line of the current LF diamond) to see if there is already a diamond there. If there is a diamond there . . . in other words, if the PEEK of location E already has a diamond (number 90) in it, then the program is told to go back to Line 20 and start over at another location. If it keeps finding lines with diamonds it will keep looking for a line which doesn't have a diamond on it because everytime it PEEKs at the end of the line it wants to start on and sees a diamond symbol there, it will start again.

Combining Movements

Let's "mix and match" a couple of movements and see what happens. Let's change the diamonds to ARROWS and keep the same LEFT TO RIGHT movement program we've been using. Then let's put BLUE HEARTS all over the screen using the A (ANYWHERE ON THE SCREEN) variable. Finally, let's say that whenever an arrow hits a blue heart, the blue heart turns RED. How in the world do we do all this? Easy. We use our new PEEK command to "look" at the screen location directly in front of the arrow as it's moving. If we PEEK ahead and see a heart, we'll change its color. Sounds simple enough in plain English, doesn't it? Let's see how it works in a program . . .

Continued next issue . . .

For Beginning Gamewriters

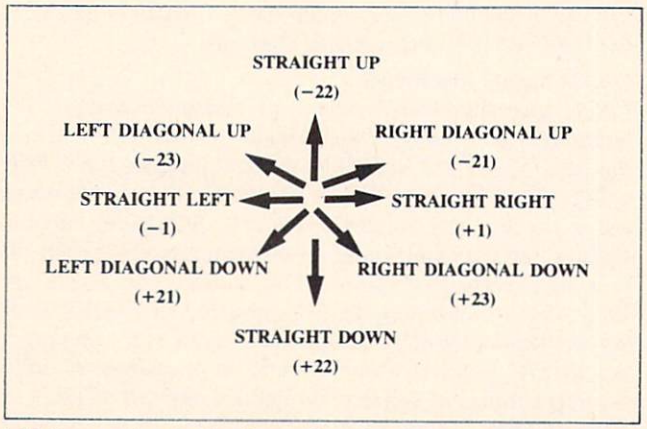
How Computer Animation Works

TWO POKE COMMANDS are required to display or move a symbol on your TV screen. Here's the format you need to use to POKE a heart in the upper right corner:

POKE 7680,83:POKE38400,5

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
7680	7681	7682	7683	7684	7685	7686	7687	7688	7689	7690	7691	7692	7693	7694	7695	7696	7697	7698	7699	7700	7701
7702	7703	7704	7705	7706	7707	7708	7709	7710	7711	7712	7713	7714	7715	7716	7717	7718	7719	7720	7721	7722	7723
7724	7725	7726	7727	7728	7729	7730	7731	7732	7733	7734	7735	7736	7737	7738	7739	7740	7741	7742	7743	7744	7745
7746	7747	7748	7749	7750	7751	7752	7753	7754	7755	7756	7757	7758	7759	7760	7761	7762	7763	7764	7765	7766	7767
7768	7769	7770	7771	7772	7773	7774	7775	7776	7777	7778	7779	7780	7781	7782	7783	7784	7785	7786	7787	7788	7789
7790	7791	7792	7793	7794	7795	7796	7797	7798	7799	7800	7801	7802	7803	7804	7805	7806	7807	7808	7809	7810	7811
7812	7813	7814	7815	7816	7817	7818	7819	7820	7821	7822	7823	7824	7825	7826	7827	7828	7829	7830	7831	7832	7833
7834	7835	7836	7837	7838	7839	7840	7841	7842	7843	7844	7845	7846	7847	7848	7849	7850	7851	7852	7853	7854	7855
7856	7857	7858	7859	7860	7861	7862	7863	7864	7865	7866	7867	7868	7869	7870	7871	7872	7873	7874	7875	7876	7877
7878	7879	7880	7881	7882	7883	7884	7885	7886	7887	7888	7889	7890	7891	7892	7893	7894	7895	7896	7897	7898	7899
7900	7901	7902	7903	7904	7905	7906	7907	7908	7909	7910	7911	7912	7913	7914	7915	7916	7917	7918	7919	7920	7921
7922	7923	7924	7925	7926	7927	7928	7929	7930	7931	7932	7933	7934	7935	7936	7937	7938	7939	7940	7941	7942	7943
7944	7945	7946	7947	7948	7949	7950	7951	7952	7953	7954	7955	7956	7957	7958	7959	7960	7961	7962	7963	7964	7965
7966	7967	7968	7969	7970	7971	7972	7973	7974	7975	7976	7977	7978	7979	7980	7981	7982	7983	7984	7985	7986	7987
7988	7989	7990	7991	7992	7993	7994	7995	7996	7997	7998	7999	8000	8001	8002	8003	8004	8005	8006	8007	8008	8009
8010	8011	8012	8013	8014	8015	8016	8017	8018	8019	8020	8021	8022	8023	8024	8025	8026	8027	8028	8029	8030	8031
8032	8033	8034	8035	8036	8037	8038	8039	8040	8041	8042	8043	8044	8045	8046	8047	8048	8049	8050	8051	8052	8053
8054	8055	8056	8057	8058	8059	8060	8061	8062	8063	8064	8065	8066	8067	8068	8069	8070	8071	8072	8073	8074	8075
8076	8077	8078	8079	8080	8081	8082	8083	8084	8085	8086	8087	8088	8089	8090	8091	8092	8093	8094	8095	8096	8097
8098	8099	8100	8101	8102	8103	8104	8105	8106	8107	8108	8109	8110	8111	8112	8113	8114	8115	8116	8117	8118	8119
8120	8121	8122	8123	8124	8125	8126	8127	8128	8129	8130	8131	8132	8133	8134	8135	8136	8137	8138	8139	8140	8141
8142	8143	8144	8145	8146	8147	8148	8149	8150	8151	8152	8153	8154	8155	8156	8157	8158	8159	8160	8161	8162	8163
8164	8165	8166	8167	8168	8169	8170	8171	8172	8173	8174	8175	8176	8177	8178	8179	8180	8181	8182	8183	8184	8185

Screen Location Chart



Add these values to the Screen location to move symbols in the direction indicated by the arrow.

There are FOUR numbers involved here: the SCREEN LOCATION (7680), POKE VALUE OF THE SYMBOL (83), COLOR LOCATION (38400) and COLOR POKE VALUE (5). Here's how they work:

1. SCREEN LOCATION. There are 506 locations, numbered from 7680 in the top righthand corner to 8185 in the bottom right corner . . . the accompanying chart shows all the location numbers.

2. POKE VALUE OF THE SYMBOL you want to display. See the chart on page 141 of your VIC owner's guide. Example: the POKE value of a solid ball is 81.

3. COLOR LOCATION. This matches the screen location but is easy to calculate because color location is ALWAYS THE SCREEN LOCATION NUMBER ADDED TO THE NUMBER 30720. Example: color location for screen location 7680 is 7680+30720 or 38400.

4. POKE VALUE OF THE COLOR you want to use. Color numbers are: 0-black, 1-white, 2-red, 3-cyan, 4-purple, 5-green, 6-blue and 7-yellow.

To POKE a purple ball in the upper right corner of the screen, type this: POKE7701,81:POKE38421,4 and press the RETURN key (or put these commands in a line of a BASIC program). POKE7701,81 means you are POKEing the number of the ball symbol (81) into screen location 7701 (see chart). POKE38421,4 means you are POKEing the color 4 into the color location that matches the screen location (38421=30720 + 7701).

Moving Symbols

To move a symbol in one of the directions shown by the arrows in the accompanying diagram, simply POKE the symbol on the screen, then add or subtract the values shown for both the SCREEN AND COLOR LOCATIONS. To simulate animation, you then ERASE the symbol from its previous position by POKEing the number 32 (a blank space) into the previous location. Type this:

POKE7910,83:POKE38630,4 (and press RETURN)

To MOVE this symbol one space down and to the right, type this:

POKE7933,83:POKE38663,4:POKE7910,32 (and press RETURN)

The first two POKEs added the value 23 to screen and color locations (as indicated by the movement diagram) and the third POKE erased the symbol by POKEing a 32 (blank space) into the previous location. Smoother, more efficient animation is achieved by using FOR . . . NEXT loops to repeatedly add or subtract a location (see examples in this article).

NEW!

The DeskTopper

$49.95
plus shipping

Holds the Commodore 64™ or VIC-20™

A beautiful oak stand that's strong and good looking. The **DeskTopper** hides all of those ugly connector cables. Your computer system will look great at home or at work. Finally, everything fits neatly on a desktop.

The **DeskTopper** holds one or two disk drives, a cassette recorder, a monitor or TV set, and a VIC-20 or 64. There's plenty of room in the back of the **DeskTopper** to hide other computer accessories: power supply, modem, RF modulator, game cartridges, memory expansion, and more.

You can get your **DeskTopper** from your local Commodore dealer, or directly from **Madison Computer.** Just call us at **608-255-5552.** Charge it on your Visa or MasterCard.

The oak **DeskTopper is only $49.95** . . . a beautiful way to organize your computer system.

VIC 20 and Commodore 64 are trademarks of Commodore Business Machines, Inc.

MC

Madison Computer 1825 Monroe, Madison, WI 53711

Call To Order
608-255-5552
Charge it on
Visa or MasterCard
Dealer Inquiries Invited

PROGRAMMER'S NOTEBOOK

Mastering Random Numbers

by
Neil Harris

There isn't much that a computer can do that people can't. There are some tasks where computers are faster, and others where people can outperform the computer.

I can name one thing off-hand that your computer can do better than you, with accuracy as well as speed.

It can create random numbers.

People use randomness in different ways. The most common is for games, but randomness is important to sociology, statistics, and quantum physics. The laws of probability tell us of likely outcomes, but we never know what actually will occur.

People are very bad ramdom number generators. Try it yourself. Take a pencil and paper, and ask a friend to give you a list of numbers between 1 and 10, at random. Stop them when they've given you 50 numbers. Then take a look at the list.

You should notice something strange. Chances are that some numbers are under-represented, or even omitted entirely. The list will usually show "clumps," certain areas where a number comes up many more times than it should by chance.

Mankind has always relied on implements to create the randomness needed for games. Decisions can be made by flipping a coin, which yields a 50% chance for heads and 50% for tails. Many games rely on dice, the most common of which have six sides, or a 1/6 chance for any side. Other random number generating devices are roulette wheels, playing cards, and the last digit of the Dow Jones stock index (although this last one is a little slow, it is conveniently public).

The computer can create very random numbers, and do it as often as you like. The RND(X) function of BASIC creates a number between 0 and 1 (*not* inclusive). This number is not truly random in the textbook sense, but is many orders of magnitude better than you or I could do. In fact, there are ways to insure that your numbers are as random as possible.

Most people don't seem to understand what the number inside the parentheses means. Programs tend to use the format of X = RND(1) without knowing why 1 works better than any other number. The number *is* significant. If the number is negative, a special "pseudo-random" sequence of numbers is started. The sequence is continued by using positive numbers in the RND function.

```
10 REM PSEUDO-RANDOM NUMBERS, BY ---> NEIL HARRIS
20 INPUT "GIVE ME A NUMBER" ; A
30 A = - ABS (A) : X = RND ( A )
40 FOR L=1 TO 10: PRINT X,
50 X = RND ( 1 ): NEXT: GOTO 20
```

If you use the same number, you get the same sequence back. This feature was designed for statistical jobs where the sequence must be repeated, but is useful for any program.

To keep your numbers as random as possible, start with the line X = RND (−TI), then use X = RND (1) throughout the program. TI is a variable that is updated every 1/60th of a second by the computer. It changes so rapidly that it is nearly impossible for a person to start the sequence from the same number twice.

The only way I know to foil this system is to type TI$="000000"; RUN., which sets the clock to 0. You might want to put the RND (−TI) later in the program, after an INPUT statement, which helps randomize the time factor.

Now we know how to make the numbers as random as possible. The next step is to mold the numbers so they come out in the "shape" you want. A fraction between 0 and 1 must be converted to a nice even number, say between 1 and 6.

It helps to think of a random number as not a specific value but a range of possible values. You might remember learning about number lines in math class; now you'll see one of the few practical uses for that lesson.

A "raw" random number would look like this on the number line:

X = RND(1)
...−1...0XXX1...2...3...4...5...6...7

It is equally probable for the value of any random number to lie anywhere between 0 and 1 on this line.

Let's suppose we want to s-t-r-e-t-c-h the number line, so we could get any number between 0 and 6. The random numbers are stretched by multiplying the random number, in this case by 6.

X = RND(1) *6
...−1...0XXXXXXXXXXXXXXXXXXXXXX6
...7...

This is almost enough to simulate dice, but not quite. Did you ever roll dice and get a 4.53424234? The INT function serves to chop off any fractional values.

X = INT(RND(1) *6)
...−1...X...X...X...X...X...6...7...

Now we're very close, the number comes up either 0, 1, 2, 3, 4 or 5. It is a simple matter to just add 1 to the result, which pushes the numbers to the right on the number line.

X = INT(RND(1) *6) +1
...−1...0...X...X...X...X...X...7...

All the values that result are equally probable, which is correct for a set of dice. This concept is vital to proper use of random numbers. Imagine in the example above that we multiplied by 5.5 instead of 6. The correct range, of 1 to 6, would result, but there would tend to be only half as many 6's as any other number.

As a measure of the probability of the random numbers, I worked up a little graphing program that works on any Commodore computer. It picks random numbers from 1 to 6, and displays the number of times each number comes up.

```
10 REM GRAPH 1 DIE, BY ---> NEIL HARRIS
20 DIM A(6): DATA 165, 180, 181, 161, 182, 170, 167, 32
30 C$ = CHR$(19): FOR L=1 TO 25: C$ = C$ + CHR$(17): NEXT
40 FOR L=0 TO 7: READ A: A$(L) = CHR$(146)
45 IF L>3 THEN A$(L) = CHR$(18)
50 A$(L) = A$(L) + CHR$(A): NEXT: PRINT CHR$(147)
60 X = INT( RND(1) *6) +1: A(X) = A(X) + 1: Q = A(X)
70 PRINT LEFT$(C$,X+1);
80 IF Q>8 THEN PRINT CHR$(18) CHR$(32);: Q = Q - 7:
   GOTO 80
90 IF Q>0 THEN PRINT A$(Q - 1)
100 GOTO 60
```

Try changing the 6 in the formula in line 60 to a 5.5 to see the effect.

When you roll 2 dice, the probabilities change. The result is a number from 2 to 12, but the numbers are *not* equally probable. The number 7 comes up 6 times as often as a 2 or a 12. Therefore, merely stretching the number line is not enough. You must pick 2 separate numbers and add them together. The formula X = INT (RND (1) *11) +2 gives results with the same range, but not the same probability, as X = INT (RND (1) *6) + INT (RND (1) *6) +2. The next two program changes graphically demonstrate this probability difference.

(Note: Use the graph program with these changes.)

```
10 REM RANDOM 2 - 12
60 X = INT( RND(1) *11) +2: A(X) = A(X) + 1: Q = A(X)
10 REM GRAPH 2 DICE
60 X = INT( RND(1) *6) + INT( RND(1) *6) + 2: A(X) = A(X)
   +1: Q = A(X)
```

The difference in likelihood between the different numbers is even more dramatic with more dice, as shown in these two changes to the program:

```
10 REM GRAPH 3 DICE
60 X=0: FOR L=1 TO 3: X=X +INT(RND(1) *6) : NEXT: A(X)=A(X)
   +1: Q=A(X)

10 REM GRAPH 10 DICE
60 X=0: FOR L=1 TO 10: X=X +INT(RND(1) *6): NEXT
65 X= INT(X/3): A(X) = A(X) +1: Q = A(X)
```

Some properties stay the same when you increase the number of dice. The graphs stay symmetrical about the center. As the number of dice increases, the graphs become steeper. If two formulas give a result with the same range, their average values are the same. What changes is the statistical value known as *standard deviation*. The numbers just become less likely to stray from the average value when there are more dice.

Just as an aside, if you don't need a nice even distribution of random numbers, you can stretch the line *after* using the INT function. INT(RND(1) *4 + 1) *3 gives numbers from 1 to 10, but omits 2, 3, 5, 6, 8, and 9, as shown in the program changes below.

```
10 REM RANDOM WITH GAPS
60 X = (INT( RND(1) *4) +1) *3: A(X) = A(X) +1: Q = A(X)
65       (delete line)
```

```
X = (INT( RND(1) *4) +1) *3

   ...-1...0...X...1...2...3...X...5...X...7...
```

That should take care of you for most situations requiring randomness, except for one. If you need to pick random numbers without repeating numbers, you must work a little harder. Imagine that you want to deal a deck of cards. You only want each particular card to come up once. Just picking random cards doesn't work unless you have a way to "use them up" once they've been displayed.

The easiest way to do this is to keep a list of all the cards, and perform a computer shuffle on them. Unlike most programs, there is only one best way to shuffle cards on a computer. First, fill an array with all of the cards. DIMension the array to 51 (which holds 52 including the 0th element). Then put a different card value in each element. In A(0) put a 0, which will symbolize the ace of hearts. In A(1) put a 1, for the deuce of hearts, and so on.

Once the deck is set up, it's time to shuffle. Just start a loop from the top card (#51) and count down to card #1. For each card, exchange it with a random card lower down in the deck. When you've reached the last card, the deck is shuffled. Here's the BASIC program to do this:

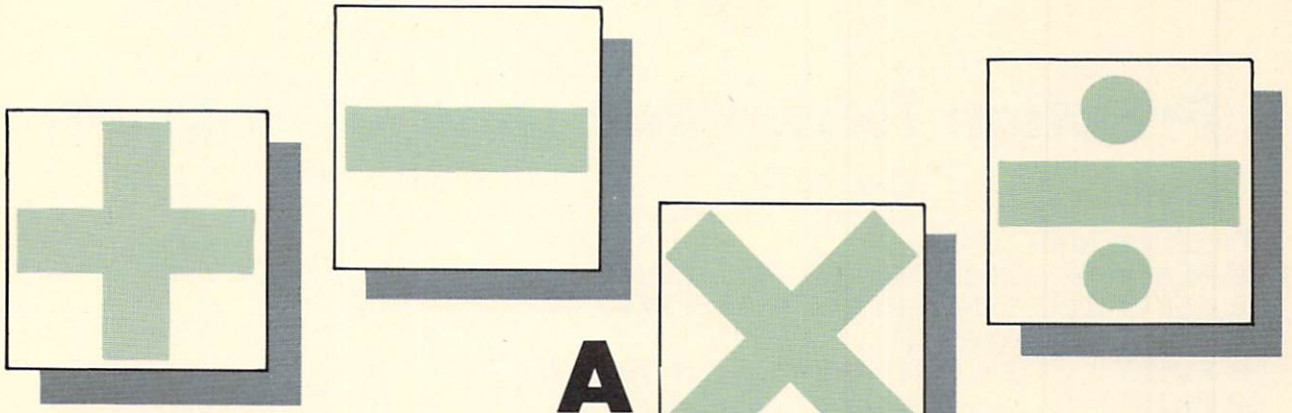
```
10 REM SHUFFLE, BY ---> NEIL HARRIS
20 DIM A(51): FOR L=0 TO 51: A(L) = L: NEXT
30 FOR L=51 TO 1 STEP -1
40 X = INT( RND(1) *L): A = A(X): A(X) = A(L): A(L) = A
50 NEXT: FOR L=0 TO 51: PRINT A(L);: NEXT
```

This, by the way, is the same routine that got me started on writing the "VIC 21: Casino Style Blackjack" program that Commodore now sells in a 6-pack.

The next step is to let the program tell which card, not just which value. Make the additions shown below and the program not only shuffles the deck, it reads the cards to you.

```
10 REM SHUFFLE WITH NAMES
50 NEXT: FOR L=0 TO 51: RESTORE
60 S = INT( A(L) / 13): C = A(L) -S *13
70 FOR M=0 TO 12: READ A$: IF M =C THEN PRINT A$" OF ";
80 NEXT: FOR M=0 TO S: READ A$: NEXT: PRINT A$,
90 NEXT L: PRINT: GOTO 30
100 DATA ACE, DEUCE, THREE, FOUR, FIVE, SIX, SEVEN, EIGHT
    NINE
110 DATA TEN, JACK, QUEEN, KING, HEARTS, DIAMONDS, CLUBS,
    SPADES
```

When dealing with random numbers, don't forget to fold, spindle, and mutilate them . . . until they do just what you want them to! C=



A BIT OF ARITHMETIC

by Elizabeth Deal

Here is a profound problem to solve in machine code: we have a number to which we may wish to add ten (or any two-byte number <32768) or from which we may wish to subtract ten with one restriction: the resulting value must be between zero and 65535 ($FFFF in hex). If it cannot fit in this range we leave the original value alone.

Let's call the original number VAL, and the 10 we'll subtract or add OFS. There is no point doing two procedures, adding a negative number takes care of subtraction, so OFS can be plus or minus 10. In BASIC we'd code:

```
TEMP=VAL:VAL=
  VAL+OFS:IF VAL<0 OR
  VAL>65535 THEN VAL=TEMP
```

Simple enough. Nothing changes if VAL+OFS cannot fit within the range. In machine code we do the same thing. Except that there is a catch. Whereas in BASIC we worked with five-byte floating numbers, here we work with two-byte integers. When we try adding/subtracting near the edges a wraparound results.

Also, and this is the tricky part, we have to think of VAL as an unsigned number between $0000 and $FFFF and we think of OFS as a signed number between $0000 and $FFFF, such as $000A for plus ten, and $FFF6 for minus ten (half the range is positive, the other half is negative). Jim Butterfield says that Humpty Dumpty said "when I use a number, it means just what I choose it to mean . . .". And that's the painful truth in machine language.

We have to think very carefully how to design one test for all of this. Second grade arithmetic comes very handy.

When we add OFS to VAL, the carry flag must remain clear at the end. If it is set, we have jumped the high end of the permitted range, and need a third byte which we don't have.

When we subtract OFS from VAL, we must not need a borrow out of a third, nonexistent, byte, so the carry flag must be set, otherwise we have jumped the low end of the range.

We're almost there. This is really a pack of fun. At this point, the tests are:

```
IF we ADDed then BCS ERROR
IF we SUBTRACTed then
  BCC ERROR
```

Handling those two lines is a bit nasty. Since we're trying for one clear indication, a bit of carry-flag and sign-of-OFS manipulation should do the job.

Knowing that we subtract when the high byte of OFS is negative ($80-FF), and we add when the high byte of OFS is positive ($00-$7F) we can, in fact have one test. Here is one possible translation of the BASIC statement:

```
1 CLC
2 LDA VAL:ADC OFS:TAX
3 LDA VAL+1:ADC OFS+1:TAY
4 ROR
5 EOR OFS+1
6 ROL:BCS DONE  ;or just BMI DONE
7 other tests on X,Y:BCS DONE
8 STX VAL:STY VAL+1
9 DONE RTS
```

Line 2 adds low bytes, keeps result in X-register. Line 3 adds the high bytes, keeps in Y. Line 4 moves the carry flag to 7th bit of accumulator (destroying A, but that's OK). At this point the number in A is positive if the carry was clear, negative if set. Line 5 is the workhorse: if the signs are the same, a zero is put in bit 7, if different, we'll see a one there, flagging an error condition defined above. So line 6 makes a decision; if all is well, line 8 replaces VAL with VAL plus or minus OFS, otherwise we exit the subroutine with VAL unchanged.

I prefer a longer version of the test in line 6, rather than a simple BMI, because of the original purpose of an unambiguous result. If before exit I were to further test VAL against other values (line 7), I could still use carry-set as an error flag.

People who have assemblers can ask the assembler to generate OFS numbers, by asking for ".WORD 10" or ".WORD −10". This is easier than thinking. C=

Fraction Reduction (and GCD-LCM)

```
 10 REM**FRACTIONS**
 20 REM**FOR  2001 SERIES  3.0 ROM**
 30 REM**EDMUND RICCHEZZA - 12/82**
 40 PRINT"■PROGRAM WILL":PRINT
 50 PRINT"    1) REDUCE FRACTION TO LOWEST
 60 PRINT"       TERMS.
 70 PRINT:PRINT"    2) PROVIDE GREATEST COMMON
 80 PRINT"       DIVISOR.
 90 PRINT:PRINT"    3) PROVIDE LEAST COMMON
 100 PRINT"       MULTIPLE.
 110 PRINT:PRINT"    ■4) NO IMPROPER FRACTIONS
 120 PRINT:PRINT"    ■5) NO > THAN 9 DIGITS
 130 FOR T=1 TO 8000:NEXT:PRINT"■"
 140 PRINT"ENTER FRACTION
 150 PRINT:INPUT"ENTER NUMERATOR
     THEN PRESS 'RETURN'";X
 160 IF X>999999999 THEN PRINT"■## TOO BIG":GOTO 150
 170 PRINT:INPUT"ENTER DENOMINATOR
     THEN PRESS 'RETURN'";Y
 180 IF Y>999999999 THEN PRINT"■## TOO BIG":GOTO 170
 190 IF X>Y THENPRINT:PRINT"■NO IMPROPER FRACTIONS":GOTO150
 200 J=X:K=Y:L=1:M=N:O=N:Q=1:K1=Y
 210 R=INT(J/K)
 220 S=K:K=J-R*K:J=S
 230 IF K=0 THEN 260
 240 T=M::M=L-R*M:L=T:U=Q:Q=O-R*Q:O=U
 250 GOTO210
 260 PRINT:PRINT
 270 IF X/Y=1 THEN PRINT"RESULT IS UNITY":PRINT:GOTO 320
 280 PRINT"LOWEST TERMS OF"X"/"Y"=":PRINTX/J"/"Y/J
 290 PRINT:PRINT"GREATEST COMMON DIVISOR:"J
 300 PRINT"LEAST COMMON MULTIPLE:"X*Y/J
 310 FOR T=1 TO 1500:NEXT
 320 PRINT:PRINT"AGAIN? 'Y' OR 'NO'
 330 GET Y$:IFY$=""THEN 330
 340 IF Y$="Y"THEN 140
 350 PRINT:PRINT"HOPE YOU ENJOYED THE EXERCISE"
READY.
```

Even the best computer needs an occasional check-up

We'll make certain your Commodore computer is healthy for years to come. We're Fisher Scientific's Instrument Service Division (ISD), a nationwide staff of factory-trained computer specialists ready and able to repair any Commodore computer unit or peripheral device. Quickly, precisely, expertly. We're part of an established, 80-year old firm with a history of servicing complex instrumentation.

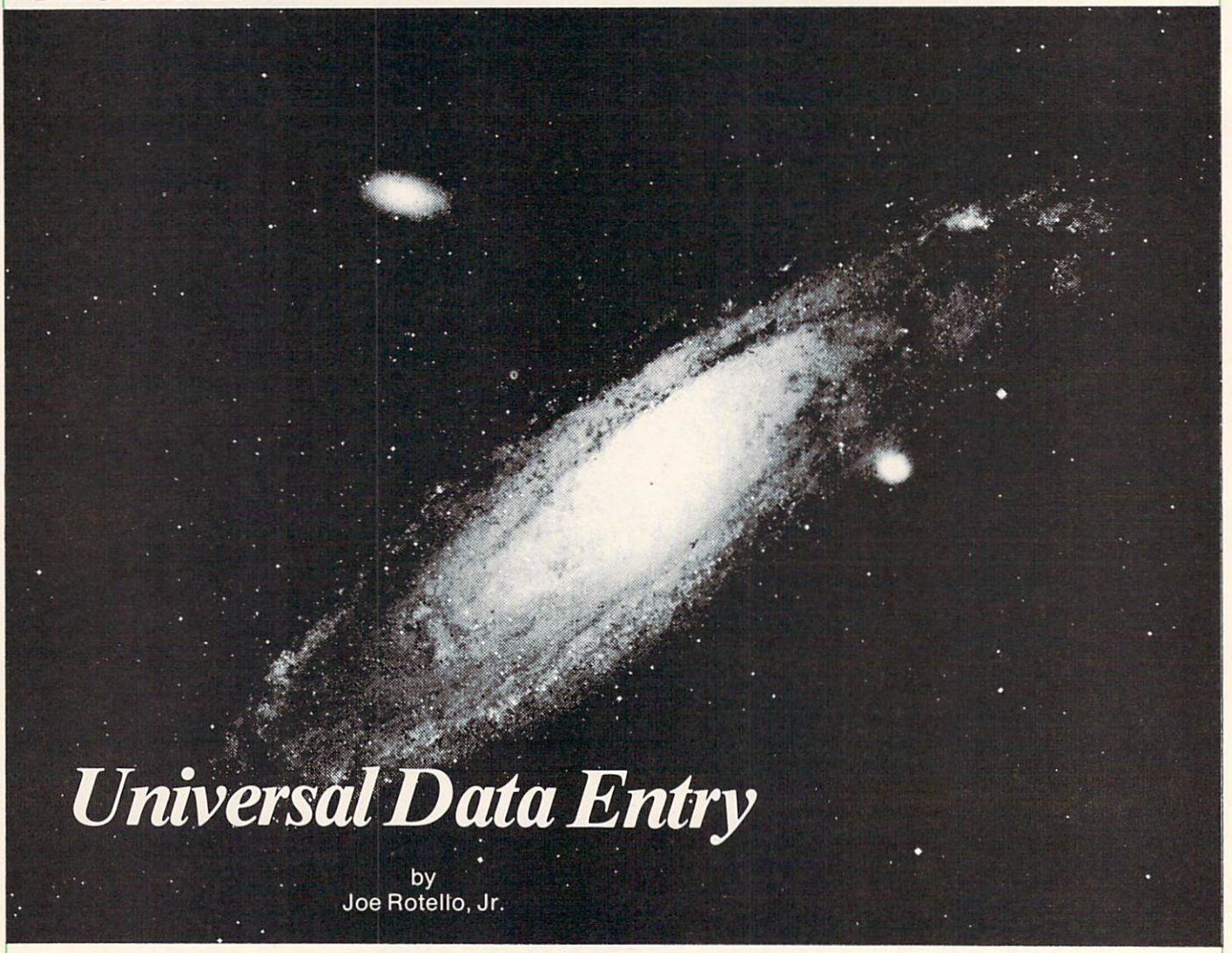
We stock needed components and parts — as close as your nearest ISD service center. And even if you didn't originally purchase your Commodore computer from Fisher, we'll repair it. We also offer a variety of preventive maintenance programs to meet your needs.

To learn more about the benefits of ISD, call the service center below closest to you. It's there to serve you.

Atlanta (404) 449-5050
Boston (617) 391-6110
Chicago (312) 773-3075
Cincinnati (513) 793-5100
Houston (713) 495-6060
Los Angeles (714) 832-9800
New York City (201) 379-1400
Orlando (305) 857-3600
Philadelphia (215) 265-0300
Pittsburgh (412) 784-2600
Raleigh (919) 876-2351
Rochester (716) 464-8900
San Francisco (408) 727-0660
St. Louis (314) 991-2400
Washington, D.C. (301) 587-7000

Fisher Scientific

An ALLIED Company



Universal Data Entry

by
Joe Rotello, Jr.

Whether you are an experienced "code banger" or a novice "key kidder" type of programmer, you eventually have to arrive at some method of data entry in your programs. We are faced with a number of methods to arrive at "fail safe" inputs, i.e., BASIC code that does not "crash" when nothing but a carriage return is input, a way to accept only certain types of data that may be keyed in by the user, to accept only so many characters, and on and on.

Would you perk up and feel better if we had what amounts to a "Universal Data Entry" (UDE for short) routine that would take care of most of our data input situations when we, or people using programs we create, have to enter data?

Gotcha interested, eh?

Ok, lets take a look at a sample UDE program that will handle most of the following requirements:

1. Will not crash BASIC if nothing but a <RETURN> is input from the keyboard, but instead, returns the user to the start of the data entry sequence.

2. Allows the programmer to easily select not only how many characters to allow to be input but also the type of characters to allow in, either alphanumerics (a-z and/or 0-9) or numerics only (0-9).

3. Allows any number of characters to be input, from as few as one to as many as can fit into our particular video screen width.

4. Within logical limits, allow you to customize the UDE routine for use in your 40-column PET/C-64, 80-column CBM or 22-column VIC.

5. Prohibits the user from entering more than a preset maximum of characters.

6. Allows the user to delete errors in the data being entered BEFORE hitting the <RETURN> key.

7. Allows the programmer to change the data header, such as "NAME" or "ADDRESS", or the type of data to be allowed, or the maximum length of data to be allowed at any point in the program and still run the same UDE subroutine.

8. Can accomplish all the above in less than twenty lines of BASIC code.

Aha! We have you drooling you say?

Now, before we start, let's set the limits of the following program. It's only an example, and as such, can undoubtedly be modified to cover different or specialized needs. So, don't expect it to be a "cure all". It's not intended to be.

The routine is in BASIC, and the variables used in the program fall as much as possible within one range of letters

so as not to become confused with other parts of your program that may include the UDE routine.

Removing the REMark statements reduces the code from about 50 lines to under 20 and the program will run faster in BASIC. The REM statements are there to act as "road maps" so we can spot the various sections of the program.

Be careful to enter the code lines EXACTLY as shown and pay special attention to such items as print statements with cursor movements and the logic lines that form the heart of the program. By the way, those who wish may compile both their user program and the UDE routine into Petspeed™.

Let's proceed to inspect the code and see what does what.

LINE(s)

80-140 define our variables and since they are REMark statements, can be deleted from the final program.

170 defines our graphic "underline" character.

180 defines our "modified cursor" and builds our maximum data length string (more on this line later).

240 a temporary line that defines our example data request parameters. In actual use, this line would be elsewhere in our BASIC program and we would define our parameters each time we wanted to define a data entry in our program.

260 builds the graphic "underline" defined in line 170. Note that the underline is automatically made the same length as the variable length "UM". In other words, if we set UM to be 25 characters, the underline will also be exactly 25 characters in length.

270 clears the video screen, prints our data header, in this case "NAME PLEASE", which we defined in line 240, a few "..." characters to make things look good, our graphic underline and positions the "cursor" at the beginning data position (first print position) above the underline.

280 examines the results of the string input by the user. If the only data input was a <RETURN>, the program returns to line 270 to begin the input cycle all over again.

330 clears the keyboard buffer so no "stray" keyboard entries enter our data request.

340 prints our "cursor" and checks to see if any data input has been made. If nothing has been input, the program loops at line 340 until something is keyed in.

350-430 These lines subject the character that is entered to a series of tests to see if the character is valid. Note that the variable UP (that we set to either 1 or 0 in line 240) is examined in lines 390 and 430. If the UP flag was set to "1", the program allows the user to input both numbers and letters. For example, input might be "Rogers 1451 West Road."

However, if the UP flag is set to "0", the program ONLY allows numbers to be input, for example, "123456". This "data test" by the UDE allows the programmer to select what type of data will be allowed. True, programmers will use alphanumeric input (a-z and 0-9) far more frequently than pure numerics (0-9). But it is important to allow the choice.

440-460 further test the data that was input from the keyboard. If the user input a <DELETE> (CHR$(20) in PET ASCII), the program erases the last character input and moves the cursor back one space. The user may erase as many characters as necessary up to the first character position. At that point, the program refuses to allow the user to erase any more.

470 checks to see if the length of the data input has reached the maximum allowed by the value of variable UM. If the values of UB and UM match, the user has "hit the limit" and the program loops to line 340 to effectively prohibit the user from entering any more characters.

480 builds the data string by adding together bytes that pass the test lines. For example, if the user input "R O G E R", by typing the appropriate keys five times, the final data string would be "ROGER".

320 double checks and forms the final output string (UO$), clears the variables used by UDE and exits the UDE routine. GOTO **** in line 320 should be replaced by the line that represents your main program, the point in your program where you will make use of the data contained in UO$.

As you can see, the program consists of fairly simple BASIC code acting in a byte-by-byte fashion on each character that is input from the keyboard.

The length limit of data input for the entire program is set at 30, by variable UM, in line 240. Why 30? Well, look at the following example, assuming you have a 40-column video screen:

NAME PLEASE . . . .

Hmm, we have a problem. The screen will seem to "overlap" because the combined length of the printed header "NAME PLEASE" and the maximum data length, thirty characters in the above example, exceeds the width of our video screen.

So, as we can see, you have to adjust the length of both UE$, in line 180, and UP$, in line 240, to suit your individual program needs and video screen width. This precaution applies especially to the VIC 20's 22-character screen width. We selected 30 as the average maximum you may use. Try to reduce the value of UM if you are using the routine on a VIC 20.

Also, remember that the data string output by the UDE routine, UO$, is just that, a STRING value. If the UDE

routine was set to allow only numerical data entry, you may have to change the string value contained in UO$ into a numerical value, using the VAL$ function, before using the UO$ output in your program.

Although the routine seems to use a lot of variables, the programmer only has to deal with three main variables once the program is properly adjusted for the particular video screen width. We also try to conserve memory by nulling out all of the unneeded variables at the end of the UDE routine when their use is no longer required.

The real power of the UDE routine comes from the fact that the program may call the same routine time and again and that the programmer has great flexibility over the parameters that are assigned to the UDE routine.

For example, the following needs would be filled by simply assigning the corresponding values to variables UP$, UP, and UM:

DATA NEEDED	UP$	UP	UM
Name, 25 characters max	NAME	1	25
Phone number, 10 numbers maximum.	NUMBER	0	10
Address, 15 characters maximum.	ADDRESS	1	15

USING THE ROUTINE

When inserting the routine into your BASIC program, be sure to put the code in lines 170 and 180 at or near the beginning of your program or, in any event, place the code out of the subroutine area. If your program keeps on executing this code by mistake, your BASIC program will eventually detect the error and crash.

Be sure to delete line 260, our "example data request", from the UDE routine and place your version of this line in the proper location(s) in your BASIC program. Also note that each time the UDE code is called, the data output string (UO$) properly erases itself at the beginning of the UDE routine, this will keep everybody straight (including the computer).

If the UDE program does not work quite right, check to make sure that the code was entered correctly, making note that you have not confused UP$ and UO$ and that you have properly set the UP flag.

Well, as we said at the beginning, this program is not a cure-all answer, but it goes a long way towards taking the pain out of programming data inputs. Feel free to take the program presented here and mold it to your needs. Perhaps you will discover other strange and wonderful uses for the Universal Data Entry program.

```
 10 REM  ************************************************************
 20 REM  ** UNIVERSAL DATA ENTRY ROUTINE AND DATA ENTRY PROGRAM **
 30 REM  **              11/08/82 - J ROTELLO                      **
 40 REM  ************************************************************
 50 REM
 60 REM   ********************************************************
 70 REM   SET VARIABLES
 80 REM   CU$ = PUSEDO FLASHING CURSOR
 90 REM   UT$ & AT$ = GRAPHIC UNDERLINE FOR DATA ENTRY
 100 REM  UE$ = MAX DATA LENGTH (30 IN THIS PROGRAM)
 110 REM  UP$ = THE UNIVERSAL STRING REQUEST NAME
 120 REM  UO$ = THE REQUESTED STRING OUTPUT
 130 REM  UP  = FLAG (0=ACCEPT NUMBERS),(1=ACCEPT ALPHANUMERICS)
 140 REM  UM  = MAX NUMBER OF CHARACTERS REQUESTED
 150 REM  ********************************************************
 160 REM
 170 UT$=CHR$(163)
 180 CU$(0)=CHR$(175):CU$(1)=" ":FORI=1TO30:UE$=UE$+"▌":NEXT
 190 REM
 200 REM  ************************************************************
 210 REM  UNIVERSAL DATA ENTRY ROUTINE
 220 REM  IMPORTANT!! THE VALUES ON THE LINE BELOW ARE SET
 230 REM  FROM THE AREA OF THE PROGRAM THAT CALLS THIS ROUTINE!!!
 240 UP$="NAME PLEASE":UP=1:UM=30:REM SAMPLE DATA REQUEST
 250 REM  ************************************************************
 260 FORI=1 TO UM:AT$=AT$+UT$:NEXT:REM BUILD UNDERLINE TO MATCH UM
     INPUT LIMIT
 270 UO$="":PRINT"□■";UP$ "....■";AT$;LEFT$(UE$,UM)"□";
 280 GOSUB330:IFAB$=""THENPRINT"□□":GOTO270:REM IF NULL INPUT DO AGAIN!!
 290 REM  ************************************************************
 300 REM  BUILD THE FINAL UO$ OUTPUT, CLEAR VARIABLES AND EXIT ROUTINE
 310 REM  ************************************************************
 320 UO$=LEFT$(AB$+AC$,UM):U1=0:UB=0:AB$="":UM=0:GOTO **** : REM EXIT
     FROM HERE
 330 AB$="":FORI=1TO10:GETA$:NEXT:REM OPTIONAL PRINT CHR$(7);
 340 PRINTCU$(-V)"▌";:FORZ=0TO9:GETU1$:IFU1$=""THENNEXT:V=NOTV:GOTO340
 350 U1=0:IFU1$>""THENU1=ASC(U1$)AND127:IFU1=13THENPRINT" ":RETURN
 360 REM ************************
 370 REM ACCEPT LETTERS & NUMBERS
 380 REM ************************
 390 IF UP=1AND(U1<32ORU1>90)ANDU1<>20ORU1=34THEN340
 400 REM ********************
 410 REM ACCEPT NUMBERS ONLY
 420 REM ********************
 430 IF UP=0AND(U1<48ORU1>57)ANDU1<>20ORU1=34THEN340
 440 UB=LEN(AB$):IFU1=20ANDUB<1THEN340
 450 IFU1=20ANDUB=1THENPRINT"■ ■■■ ■▌";:GOTO330
 460 IFU1=20THENAB$=LEFT$(AB$,UB-1):PRINT"■ ■■■ ■▌";:GOTO340
 470 IFUB=UMTHEN340
 480 AB$=AB$+U1$:PRINTU1$;:IFUM=1THENPRINT" ":RETURN
 490 GOTO340
READY.
```

Cursor Input Program

By Joe Rotello

One of the basics, no pun intended, of programming is the art of developing the program menu; the part of the program that gives the user a list of options that may be selected.

Many basic menus exist. Some menus have options that are selected by number or letter, for example:

(1) Enter Accounts Receivable
(2) Enter Accounts Payable
(3) Exit program

In these and similar menus, the user selects and keys in the proper number (or letter) and is generally followed by the <RETURN> key to complete the selection.

Our so-called "cursor input" menu method may be more beneficial under many circumstances. This menu variation seems to be easier to sell to the first time computer user, or where non-computerists, who may be unfamiliar with computers and keyboards, have to enter menu selections.

Code-wise, the program prints a solid non-blinking "cursor" in a series of vertical "boxes", one box for each menu option. By simply hitting the <SPACE> bar, the user can position the "cursor" at the proper spot and enter that menu option by pressing <RETURN>. By removing the REMark's, the code reduces to a very few lines.

This method has been put to the test by library users and serious non-computerists and seems to work quite well. The code should be suitable for any Commodore computer model.

To get us started, the sample program allows for four menu options and, by following the listing explanations, the program can be customized to your particular use. Be sure to pay special attention to the cursor movements required to vary the number of menu entries. The required changes are very simple and easy to make and the code is shown for both four and five menu entries. Compare the two and the pattern of changes, say to make two menu entries or six menu entries, will become clear.

Test for programmers: How would you change the program to space the menu entries TWO lines apart instead of as they are now? (Hint: It's simpler than you think!).

Happy menuing!

```
5 rem       version 4.1
15 rem      11/22/82
25 rem      <space> key to position cursor
35 rem      cursor input with four function choices 1 thru 4
45 rem      11/22/82
55 rem      selection entered with <return> key
65 rem      (c) 1982 by compusystems management
80 rem
85 rem      define variables
90 cr$=chr$(13):sp$=chr$(32):lc=0
95 rem      lc = line counter
105 rem     print page instruction and menu
110 print"■                                                    ■Cursor Input■■"
120 print"■■Press <■ SPACE ■> key to move cursor"
130 print"■■Press <■ RETURN ■> to enter selection"
140 print"■■■■ |  Do  Function #1"
150 print "+ |  Do  Function #2"
160 print "+ |  Do  Function #3"
170 print "+ |  Do  Function #4"
175 rem     move cursor up and in to first menu line
180 print "■■■■■";
185 rem     check for keyboard entry
190 getc$
215 rem     if no input, print cursor & do again
220 ifc$=""thenprint"■ ■";:goto190
```

```
225 rem      if a input, turn cursor off
230 ifc$=sp$thenprint"■ ■";
235 rem      if keyboard entry <return> then thats selected entry
240 ifc$=cr$then300
245 rem      let user select - move cursor as required & do again
250 ifc$=sp$andlc=0thenlc=1:print"■";:goto190
260 ifc$=sp$andlc=3thenlc=0:print"■■■";:goto190
270 ifc$=sp$andlc<>0thenlc=lc+1:print"■";:goto190
280 ifc$=sp$andlc<>3thenlc=lc+1:print"■";:goto190
285 rem      if input not a <space>, do again
290 goto190
295 rem      go to routine user has selected
300 print"■■■■■":onlc+1goto310,320,330,340
310 print" ok -  function #1":end
320 print" ok -  function #2":end
330 print" ok -  function #3":end
340 print" ok -  function #4":end
ready.
```

CODE CHANGES REQUIRED TO PERFORM FIVE MENU ENTRY:

ADD THE FOLLOWING LINES:

Line 172 - print"[ ] Do Function #5"

Line 350 - print" ok - function #5":end

CHANGE THE FOLLOWING LINES TO THE FOLLOWING CODE:

Line 180 - print"<five cursor up><cursor left>";

Line 260 - ifc$=sp$andlc=4thenlc=0:print"<4 cursor up>";goto190

Line 280 - ifc$=sp$andlc<>4thenlc=lc+1:print"<1 cursor down>";
:goto190

Line 300 - Add ",350" to end of line 300

Machine Language Disk Error Routine

For PET/CBM (Upgrade or 4.0 ROM's),
VIC 20 or Commodore 64 computers.
by
Thomas Henry
Transonic Laboratories, 249 Norton Street
Mankato, MN 56001 507/387-1642

There's no worse feeling in this world than seeing the disk error LED light up and not being able to do anything about it! For example, the editor program, which is part of the Commodore Assembler Development Package, occasionally freezes up when asked to find a source file on disk that doesn't actually exist. (A name may have been mispelled, the wrong disk might be in the drive, etc.) When this happens, the error LED lights, the computer locks up, and anything in the computer is lost. The only way to recover from this is to power down and up again! Wouldn't it be much nicer if a disk error message was printed and control was returned to the user?

It's important to assume, in any type of programming, that the user may make such mistakes. To qualify as "user friendly," a computer shouldn't let such scenarios as that described above happen. To prevent things like that from on happening Commodore equipment, the following program was written. The program not only fixes the editor program mentioned above, but more importantly, can be added to any machine language program you may have. By changing the equates as shown, you will be able to get the program up and running on PET/CBM computers with Upgrade or 4.0 ROM's, the VIC 20 or the Commodore 64.

It's easy to provide disk error protection in BASIC programs; you just OPEN the error channel and read the status. But how do you do this in machine language? As it turns out it's quite easy to do, using just a handful of ROM routines already in your computer. Before looking at the assembler listing, let's define in very precise terms just what the program should do.

We want a routine that can be called by a JSR at any time. Typically, you would jump to this routine whenever a file has been opened. If the file opened successfully, control should return to the calling routine. If trouble was encountered (FILE NOT FOUND, WRITE PROTECT ON, FILE EXISTS, etc.), then the program should direct control to some sort of error routine. For example, the error routine might send you back to a warm start, or some other "graceful" exit function. So, when the disk error routine is summoned, if everything is okay no message is printed and an RTS returns control back to the calling program. If an error is detected, default devices are restored, the error message printed, and control is diverted to an error handling routine.

Having define what the program should do, we can examine the listing. Refer to it now. There are three sets of equates; pick the set that corresponds to your computer. Note that VIC 20 and Commodore 64 users share the same set of equates (except for one location). This is due to the fact that these two computers use the same jump table.

To start the routine off, the error channel is opened by sending a talk with secondary address of fifteen ($F = $0F OR $60). The disk drive will now spew out the error message, and the routine at label XFER picks off this message. Note that we accept the message whether it is good or bad news; later on we'll decide whether to print it or not.

The message is stored, character by character in a buffer. The tape buffer was used in this instance, but any other section of unused memory that's handy may be used.

A carriage return ($0D signifies that the end of the message has been found. The carriage return is stored, and then is followed by a zero byte. By doing this, we can call the routine PSTRNG later later on, which will print out any message up to a zero byte. But more about that later.

At this point, an UNTALK is sent to the disk drive, and this clears it for further use. Next the program checks to see if the first two bytes in the message are ASCII zeros ($30). If they are, then no error has occurred, and we return to the calling routine immediately, with no message printed.

If, however, these zeros aren't found, then an error must have occurred. The devices are restored to their default values (keyboard and screen), and the message is printed out via PSTRNG. Next the stack is pulled twice, and this disables the return to the calling program. Finally a jump is made to the error routine mentioned above.

You can see, then, that it is actually quite easy to provide disk error detection in machine language. The routine, as presented in the listing, is a mere sixty-four bytes long, surely a small price to pay for "user friendliness"!

Even if you don't have an immediate use for this subroutine, it's recommended that you put it into your programming notebook for later use. The time will come sooner or later when you'll want to safeguard a program against disk failures. You can use this routine in programs of your own devising or add it to commercial programs. Either way, you'll welcome the protection from disk or computer lock-ups!

This routine has been implemented on a number of programs over the past year and has proved to be quite reliable. It has been used on both the CBM-8032 with a 4040 disk drive and the VIC 20 with a 1540 disk unit. In all cases, much hair pulling has been prevented! Here's hoping that your programs will never again suffer from disk drive errors!

```
LINE# LOC   CODE        LINE

00001  0000             ;****************************************
00002  0000             ;*                                      *
00003  0000             ;*  GENERAL PURPOSE DISK ERROR ROUTINE  *
00004  0000             ;*       FOR COMMODORE COMPUTERS        *
00005  0000             ;*                                      *
00006  0000             ;*            THOMAS HENRY              *
00007  0000             ;*         TRANSONIC LABORATORIES       *
00008  0000             ;*           249 NORTON STREET          *
00009  0000             ;*           MANKATO, MN 56001          *
00010  0000             ;*                                      *
00011  0000             ;****************************************
00012  0000             ;
00013  0000             ;
00014  0000             ;*** EQUATES FOR UPGRADE ROM PET'S ***
00015  0000             ;
00016  0000             ; DEVICE = $D4            ;CURRENT DEVICE.
00017  0000             ; BUFFER = $027A          ;TEMPORARY BUFFER FOR STRING.
00018  0000             ; PSTRNG = $CA1C          ;PRINT STRING AIMED BY (A,Y).
00019  0000             ; TKSA   = $F128          ;SECONDARY AFTER TALK.
00020  0000             ; ACPTR  = $F18C          ;ACCEPT DATA FROM DEVICE.
00021  0000             ; UNTALK = $F17F          ;UNTALK THE DEVICE.
00022  0000             ; TALK   = $F0B6          ;COMMAND DEVICE TO TALK.
00023  0000             ; RESTOR = $FFCC          ;RESTORE DEFAULT DEVICES.
00024  0000             ;
00025  0000             ;
00026  0000             ;*** EQUATES FOR 4.0 ROM PET'S ***
00027  0000             ;
00028  0000             DEVICE = $D4            ;CURRENT DEVICE.
00029  0000             BUFFER = $027A          ;TEMPORARY BUFFER FOR STRING.
00030  0000             PSTRNG = $BB1D          ;PRINT STRING AIMED BY (A,Y).
00031  0000             TKSA   = $F143          ;SECONDARY AFTER TALK.
00032  0000             ACPTR  = $F1C0          ;ACCEPT DATA FROM DEVICE.
00033  0000             UNTALK = $F1B6          ;UNTALK THE DEVICE.
00034  0000             TALK   = $F0D2          ;COMMAND DEVICE TO TALK.
00035  0000             RESTOR = $FFCC          ;RESTORE DEFAULT DEVICES.
00036  0000             ;
00037  0000             ;
00038  0000             ;*** EQUATES FOR VIC-20 AND COMMODORE 64 ***
00039  0000             ;
00040  0000             ; DEVICE = $BA            ;CURRENT DEVICE.
00041  0000             ; BUFFER = $033C          ;TEMPORARY BUFFER FOR STRING.
00042  0000             ; PSTRNG = $CB1E          ;PRINT STRING AIMED BY (A,Y).
00043  0000             ;                         ;CHANGE TO $AB1E FOR THE 64.
00044  0000             ; TKSA   = $FF96          ;SECONDARY AFTER TALK.
00045  0000             ; ACPTR  = $FFA5          ;ACCEPT DATA FROM DEVICE.
00046  0000             ; UNTALK = $FFAB          ;UNTALK THE DEVICE.
00047  0000             ; TALK   = $FFB4          ;COMMAND DEVICE TO TALK.
00048  0000             ; RESTOR = $FFCC          ;RESTORE DEFAULT DEVICES.
00049  0000             ;
00050  0000             ;
00051  0000                     * = $5000
00053  5000             ;
00054  5000  A9 08       DS     LDA #$08         ;GET READY TO READ
00055  5002  85 D4              STA DEVICE       ;THE ERROR CHANNEL.
00056  5004  20 D2 F0           JSR TALK         ;COMMAND DEVICE TO TALK.
00057  5007  A9 6F              LDA #$6F         ;THIS IS THE ERROR CHANNEL.
00058  5009  20 43 F1           JSR TKSA         ;SECONDARY ADDR. AFTER TALK.
00059  500C  A0 00              LDY #$00
```

```
LINE# LOC   CODE        LINE

00060  500E  20 C0 F1   XFER   JSR ACPTR       ;GET BYTE FROM BUS.
00061  5011  99 7A 02          STA BUFFER,Y    ;SAVE IT IN BUFFER.
00062  5014  C8               INY
00063  5015  C9 0D           CMP #$0D        ;LOOK FOR CARRIAGE RETURN.
00064  5017  D0 F5           BNE XFER        ;IF NOT, GET NEXT CHAR.
00065  5019  A9 00           LDA #$00        ;PUT ZERO BYTE FOR END.
00066  501B  99 7A 02          STA BUFFER,Y
00067  501E  20 B6 F1          JSR UNTALK      ;UNTALK THE CHANNEL.
00068  5021  A0 00            LDY #$00
00069  5023  A9 30            LDA #'0
00070  5025  D9 7A 02          CMP BUFFER,Y    ;CHECK FOR ASCII '0'.
00071  5028  D0 06            BNE BAD         ;IF NOT, MUST BE ERROR.
00072  502A  C8               INY
00073  502B  D9 7A 02          CMP BUFFER,Y    ;CHECK FOR ANOTHER ASCII '0'.
00074  502E  F0 0F            BEQ GOOD        ;IF FOUND THEN NO ERROR.
00075  5030  20 CC FF   BAD    JSR RESTOR      ;WE FOUND AN ERROR SO,
00076  5033  A9 7A            LDA #<BUFFER    ;RESTORE DEFAULT DEVICES.
00077  5035  A0 02            LDY #>BUFFER
00078  5037  20 1D BB          JSR PSTRNG      ;PRINT THE ERROR MESSAGE.
00079  503A  68               PLA             ;DON'T RETURN IF BAD, BUT
00080  503B  68               PLA             ;JUMP TO USER DEFINED
00081  503C  4C 40 50          JMP ERROR       ;ERROR ROUTINE.
00082  503F  60         GOOD   RTS             ;OKAY TO RETURN AND CONTINUE.
00083  5040               ;
00084  5040               ;
00085  5040  00           ERROR  BRK
00086  5041               ;
00087  5041               ;THIS IS THE USER DEFINED ERROR ROUTINE.
00088  5041               ;FOR EXPERIMENTAL PURPOSES YOU MAY WANT
00089  5041               ;TO LEAVE THIS 'BRK' INSTRUCTION IN.
00090  5041               ;FOR YOUR FINAL PROGRAM, REPLACE IT
00091  5041               ;WITH AN ERROR ROUTINE THAT SUITS YOUR NEEDS.
00092  5041               ;FOR EXAMPLE, YOU MAY WANT THE ERROR ROUTINE
00093  5041               ;TO BE A JUMP TO A WARM START, ETC.
00094  5041                      .END

ERRORS = 00000

END OF ASSEMBLY
```

WORD COUNT

one
two
three
four
five
six
seven
eight
nine

By David A. Hook, Barrie Ontario
Reprinted from **The Transactor**

Purpose:
After slaving over the composition of an article, most writers are required to count the words, as the basis for payment for their work. I am told that many commerical word-processors include this function. Neither WordPro 3 nor WordPro 4 contain this feature.

Although my writing efforts are infrequent, my wife has done a lot of freelance work. Currently she is working on a complete rewrite of a BASIC test to be used in Grade 9. This project involves a 40% reduction in word count. Thus, this program was created.

An initial effort was accomplished using BASIC. For a WordPro file with 2200 words, the time to perform the count was a shade over 21 minutes. This was acceptable, since other tasks (non-computer) could be performed while the CBM was busy.

However, we've all heard the praises sung for the speed of Machine Language. The logic aspect was fairly straightforward and already de-bugged in BASIC. The results are before you in this article.

The same WordPro file was counted in 12.67 seconds!!

The program works with either WordPro 3 or Word Pro 4 files and with BASIC 2.0 and BASIC 4.0 (Regular-, Fat-40 and 80-column machines). The WordPro file is read from Drive 0 of the disk unit. DOS 2.1 is not necessary, although I have not included an error-checking routine (except for BASIC 4.0).

Procedure:
First, type in the BASIC listing exactly as given below. Be very careful to include all the spaces specified, especially in line 8 of the program. There is one after the CLR/HOME, 13 before the title and 12 following.

Now SAVE this part as "WC.BAS". After VERIFYing, reset the machine for the next step:

For those who wish their own assembly, skip to step "b" below.

a) For the "non-assembler" crowd here's the method for you. Type in 'SYS4' to get into the M.L. monitor. Then enter the following line, right after the displayed "'" (at the present cursor position):

.M 0624 06BC <RETURN>

The screen will fill with a display much like that shown in the "HEX DUMP' listing below. Your task is to carefully change all of the displayed figures to match the listing (top half). Simply type in the proper values, remembering to hit 'RETURN' at the end of each line.

For the remainder, do the same again after typing this line:

.M 06BC 0733 <RETURN>

After making the required changes, this should be SAVEd, using the monitor, as follows:

.S "0:WC.ML",08,0624,0733
<RETURN>
.X <RETURN> (exit the monitor)

You may VERIFY this normally, if you wish.

Now skip to step "c" below.

b) The source code for the program has been included. This code will work with either MAE or ASM/TED assemblers.

If you choose to relocate the machine-language "start address", remember that there are three references in the BASIC portion. Be sure that these get corrected, too.

c) If you're still with me, only two things remain to be done:

Simply reLOAD "WC.BAS" first, then reLOAD "WC.ML". Use the normal BASIC SAVE command now, and both pieces will be linked together.

Remember that any changes to the BASIC portion now will also move the machine language. Do so at your own risk.

Operation:
Before you RUN the program, be sure you know the file name of the WordPro file to be counted. Put this diskette into Drive \$0, and you are ready to go.

The program self-adjusts for 40- or 80-column operation. This assumes that you will only be counting 40-column files on a 40-column machine, and 80-column files on an 80-column machine. Thus, the correction is based on the machine in use, not the file being read.

The program ignores WordPro format commands (and anything on the same line as a format command).

If you used the "--" character as a dash, there should be no preceding or following blanks. If you use a series of "--", (as I sometimes do for underlining) the count may not be correct.

If you've entered everything correctly, the word count total should have appeared on the screen, after 2-25 seconds. Disk activity should end and the "READY" prompt should now be displayed.

Since none of us ever make any mistakes, (??), you should be ready to count every WordPro file within reach. In our house, this program has had a real workout. I hope it proves useful to you too.

This is the usual place to acknowledge Jim Butterfield. I blame him for getting me into this all-consuming habit...er hobby!

WORD COUNT 9 LISTING

```
 0 REM    WORD COUNT 9 -- WORDPRO 3
 1 REM     AS OF NOVEMBER 29, 1981
 2 REM
 3 REM  (C) DAVID A. HOOK, 58 STEEL ST.
 4 REM BARRIE, ONTARIO, CANADA, L4M 2E9
 5 REM
 6 REM        ALL RIGHTS RESERVED
 7 REM
 8 PRINT"□ ₪                WORD COUNT ML
 9 PRINT"■■■PLACE PROGRAM DISK IN DRIVE #0
 10 PRINT"■■HIT A KEY WHEN READY  ■";
 11 GETZ$:IFZ$=""THEN11
 12 PRINT"  OK"
 13 INPUT"■PROGRAM NAME   ■■■■";F$
 14 OPEN1,8,15,"I0":CLOSE1
 15 OPEN2,8,2,"0:"+F$+",P,R"
 16 IFDSTHENZ$=DS$:GOTO21
 17 SYS1582
 18 PRINT"■■■■■■■■■■■■WORD COUNT = ";
 19 PRINTPEEK(1572)+256*PEEK(1573)
 20 Z$="DONE"
 21 PRINT"■■"Z$:CLOSE2:END
READY.
```

WORD COUNT 9 HEX DUMP

```
C*
     PC   IRQ   SR AC XR YR SP
.;  D780 E455 34 33 38 36 FA
.
.:  0624 45 01 99 22 11 12 22 5A
.:  062C 24 3A A2 09 A9 00 9D 24
.:  0634 06 CA 10 FA A9 28 A2 60
.:  063C 8E 00 84 AE 00 80 E0 60
.:  0644 F0 01 0A 8D 28 06 A2 02
.:  064C 20 C6 FF 20 06 07 20 06
.:  0654 07 A2 00 8E 2B 06 8E 29
.:  065C 06 8E 27 06 18 AD 24 06
.:  0664 6D 26 06 8D 24 06 AD 25
.:  066C 06 69 00 8D 25 06 8E 26
.:  0674 06 AE 27 06 EC 28 06 F0
.:  067C D8 20 06 07 EE 27 06 AD
.:  0684 2C 06 AC 2D 06 A2 00 C9
.:  068C 7A D0 09 8E 26 06 20 21
.:  0694 07 4C 55 06 C9 1F D0 0B
.:  069C AE 2B 06 F0 03 EE 26 06
.:  06A4 4C 92 06 AE 29 06 D0 21
.:  06AC AE 27 06 CA D0 13 C0 20
.:  06B4 F0 0F C0 6F F0 0B C9 20
.:  06BC F0 04 C9 6F D0 03 EE 26
.:  06C4 06 C9 20 F0 AC C9 6F F0
.:  06CC A8 A2 FF 8E 29 06 C9 20
.:  06D4 F0 16 C9 6F F0 12 C9 2D
.:  06DC D0 04 C0 2D F0 0F 8E 2B
.:  06E4 06 E8 8E 2A 06 4C 75 06
.:  06EC AE 2A 06 D0 84 AE 2B 06
.:  06F4 F0 08 EE 26 06 A2 00 8E
.:  06FC 2B 06 A2 FF 8E 2A 06 4C
.:  0704 75 06 AE 2C 06 8E 2D 06
.:  070C 20 E4 FF D0 02 09 40 8D
.:  0714 2C 06 A5 96 F0 06 20 CC
.:  071C FF A2 F8 9A 60 AE 28 06
.:  0724 CA 86 B4 20 06 07 A6 B4
.:  072C CA EC 27 06 B0 F3 60 44
.
```

```
0001              .LS
0002  ;*************************************
0003  ;*                                   *
0004  ;*     WORDCOUNT.SRC9 -- WORDPRO 3   *
0005  ;*                                   *
0006  ;*        AS OF NOVEMBER 29, 1981    *
0007  ;*                                   *
0008  ;*  (C) DAVID A. HOOK, 58 STEEL STREET *
0009  ;*       BARRIE, ONTARIO  L4M 2E9     *
0010  ;*         CANADA (705) 726-8126      *
0011  ;*                                   *
0012  ;*          ALL RIGHTS RESERVED      *
0013  ;*                                   *
0014  ;*************************************
0015  ;
0016  ; VARIABLES
0017  ;
0018  CHANNEL     .DE $02       ;DISK CHANNEL NUMBER
0019  ENDLIN      .DE $1P       ;END OF LINE
0020  BLANK       .DE $20       ;BLANK
0021  LENGTH      .DE $28       ;NORMAL = 40 CHARS.
0022  DASH        .DE $2D       ;SINGLE DASH
0023  SHFSPC      .DE $60       ;SHIFTED SPACE
0024  FORSPC      .DE $6F       ;FORCED SPACE
0025  FORCMD      .DE $7A       ;FORMAT COMMAND
0026  ST          .DE $96       ;STATUS BYTE
0027  SAVX        .DE $B4       ;KEEP R(X)
0028  ;
0029  SCREEN      .DE $8000     ;SCREEN MEMORY
0030  IMAGES      .DE $8400     ;SCREEN IMAGES (40 COL.)
0031  ;
0032  ; BASIC ROUTINES
0033  ;
0034  SETINP      .DE $FFC6     ;SET INPUT DEVICE
0035  CLRCHN      .DE $FFCC     ;RESTORE DEFAULT I/O DEV.
0036  WRT         .DE $FFD2     ;PRINT CHARACTER
0037  GETCHR      .DE $FFE4     ;GET CHARACTER
0038  ;
0039              ;.OS      (DON'T STORE CODE)
0040  ;
0041              .BA $0624
0042  ;
  24-             0043  WORDTOT     .DS 2         ;# WORDS (TOTAL)
  26-             0044  LINETOT     .DS 1         ;# WORDS (CURRENT LINE)
  27-             0045  CHARTOT     .DS 1         ;# CHARACTERS (CUR. LINE)
  28-             0046  LINLEN      .DS 1         ;LENGTH OF WORDPRO LINE
                  0047  ;
  29-             0048  LINFLG      .DS 1         ;LINE START FLAG
  2A-             0049  BLNKFLG     .DS 1         ;BLANK FLAG
  2B-             0050  WORDFLG     .DS 1         ;WORD FLAG
                  0051  ;
  2C-             0052  CURCHAR     .DS 1         ;CURRENT CHARACTER
  2D-             0053  LASTCHAR    .DS 1         ;LAST CHARACTER
                  0054  ;
  2E- A2 09       0055  START       LDX #LASTCHAR-WORDTOT    ;INITIALIZE LOC
  30- A9 00       0056              LDA #0
  32- 9D 24 06    0057  LOOP        STA WORDTOT,X
0635- CA          0058              DEX
0636- 10 FA       0059              BPL LOOP
                  0060  ;
0638- A9 28       0061  SETLEN      LDA #LENGTH  ;40/80 COLUMN ?
063A- A2 60       0062              LDX #SHFSPC
063C- 8E 00 84    0063              STX IMAGES
063F- AE 00 80    0064              LDX SCREEN
0642- E0 60       0065              CPX #SHFSPC
0644- F0 01       0066              BEQ FORTY
                  0067  ;
0646- 0A          0068  EIGHTY      ASL A
0647- 8D 28 06    0069  FORTY       STA LINLEN
                  0070  ;
064A- A2 02       0071  SETCHN      LDX #CHANNEL      ;SET CHANNEL FOR INPUT
064C- 20 C6 FF    0072              JSR SETINP
                  0073  ;
064F- 20 06 07    0074  LOADADR     JSR GET       ;IGNORE LOAD ADDRESS
0652- 20 06 07    0075              JSR GET
                  0076  ;
0655- A2 00       0077  LINESTRT    LDX #0        ;START A WORDPRO LINE
0657- 8E 2B 06    0078              STX WORDFLG
```

```
065A- 8E 29 06  0079            STX LINFLG
065D- 8E 27 06  0080            STX CHARTOT
                0081 ;
0660- 18        0082 ADDLINE    CLC  ;SUM PREV. LINE INTO TOTAL
0661- AD 24 06  0083            LDA WORDTOT
0664- 6D 26 06  0084            ADC LINETOT
0667- 8D 24 06  0085            STA WORDTOT
066A- AD 25 06  0086            LDA WORDTOT+1
066D- 69 00     0087            ADC #0
066F- 8D 25 06  0088            STA WORDTOT+1
                0089 ;
0672- 8E 26 06  0090            STX LINETOT
                0091 ;
0675- AE 27 06  0092 CHKLINE    LDX CHARTOT  ;IS LINE DONE ?
0678- EC 28 06  0093            CPX LINLEN
067B- F0 D8     0094            BEQ LINESTRT
                0095 ;
067D- 20 06 07  0096            JSR GET
0680- EE 27 06  0097            INC CHARTOT
0683- AD 2C 06  0098            LDA CURCHAR
0686- AC 2D 06  0099            LDY LASTCHAR
0689- A2 00     0100            LDX #0
068B- C9 7A     0101            CMP #PORCMD  ; WORDPRO FORMAT COMMAND ?
068D- D0 09     0102            BNE NOTFORMAT
                0103 ;
068F- 8E 26 06  0104 FORMAT     STX LINETOT  ;ZERO LINE COUNT
0692- 20 21 07  0105 FINISH     JSR GETREST  ;IGNORE REST OF LINE
0695- 4C 55 06  0106            JMP LINESTRT
                0107 ;
0698- C9 1F     0108 NOTFORMAT  CMP #ENDLIN  ;END OF LINE SYMBOL ?
069A- D0 0B     0109            BNE MORE
                0110 ;
069C- AE 2B 06  0111            LDX WORDFLG
069F- F0 03     0112            BEQ DONELINE
                0113 ;
06A1- EE 26 06  0114            INC LINETOT  ;GOT A WORD
                0115 ;
06A4- 4C 92 06  0116 DONELINE   JMP FINISH
                0117 ;
06A7- AE 29 06  0118 MORE       LDX LINFLG   ;STARTED LINE YET ?
06AA- D0 21     0119            BNE CONTLIN
                0120 ;
06AC- AE 27 06  0121 LEADBLK    LDX CHARTOT  ;LEAD BLANK IMPORTANT ?
06AF- CA        0122            DEX
06B0- D0 13     0123            BNE NOTLEAD  ;NOT FIRST CHAR.
                0124 ;
06B2- C0 20     0125            CPY #BLANK   ;LAST OF PREV. LINE ?
06B4- F0 0F     0126            BEQ NOTLEAD
                0127 ;
06B6- C0 6F     0128            CPY #FORSPC
06B8- F0 0B     0129            BEQ NOTLEAD
                0130 ;
06BA- C9 20     0131            CMP #BLANK   ;CURRENT CHARACTER ?
06BC- F0 04     0132            BEQ COUNT
                0133 ;
06BE- C9 6F     0134            CMP #FORSPC
06C0- D0 03     0135            BNE NOTLEAD
                0136 ;
06C2- EE 26 06  0137 COUNT      INC LINETOT  ;LEAD BLANK MEANS A WORD
                0138 ;
06C5- C9 20     0139 NOTLEAD    CMP #BLANK   ;IGNORE LEAD BLANKS
06C7- F0 AC     0140            BEQ CHKLINE  ;CONTINUE THE LINE
                0141 ;
06C9- C9 6F     0142            CMP #FORSPC
06CB- F0 A8     0143            BEQ CHKLINE  ;CONTINUE THE LINE
                0144 ;
06CD- A2 FF     0145 CONTLIN    LDX #$FF     ;START THE LINE
06CF- 8E 29 06  0146            STX LINFLG
                0147 ;
06D2- C9 20     0148            CMP #BLANK
06D4- F0 16     0149            BEQ WORDCOUNT
                0150 ;
06D6- C9 6F     0151            CMP #FORSPC
06D8- F0 12     0152            BEQ WORDCOUNT
                0153 ;
06DA- C9 2D     0154            CMP #DASH    ; '--' IS ALSO A WORD END
06DC- D0 04     0155            BNE NOTDASH
                0156 ;
```

```
06DE- C0 2D     0157            CPY #DASH
06E0- F0 0F     0158            BEQ DASHCOUNT
                0159 ;
06E2- 8E 2B 06  0160 NOTDASH    STX WORDFLG
06E5- E8        0161            INX
06E6- 8E 2A 06  0162            STX BLNKFLG
06E9- 4C 75 06  0163            JMP CHKLINE   ;CONTINUE THE LINE
                0164 ;
06EC- AE 2A 06  0165 WORDCOUNT  LDX BLNKFLG   ;FOUND END OF WORD ?
06EF- D0 84     0166            BNE CHKLINE   ;CONTINUE THE LINE
                0167 ;
06F1- AE 2B 06  0168 DASHCOUNT  LDX WORDFLG   ;WERE WE ON A WORD ?
06F4- F0 08     0169            BEQ NOTYET
                0170 ;
06F6- EE 26 06  0171            INC LINETOT   ;COUNT A WORD
06F9- A2 00     0172            LDX #0
06FB- 8E 2B 06  0173            STX WORDFLG
                0174 ;
06FE- A2 FF     0175 NOTYET     LDX #$FF      ;MARK THE BLANK
0700- 8E 2A 06  0176            STX BLNKFLG
0703- 4C 75 06  0177            JMP CHKLINE   ;CONTINUE THE LINE
                0178 ;
0706- AE 2C 06  0179 GET        LDX CURCHAR   ;GET A CHARACTER
0709- 8E 2D 06  0180            STX LASTCHAR
070C- 20 E4 FF  0181            JSR GETCHR
070F- D0 02     0182            BNE NONZERO
                0183 ;
0711- 09 40     0184            ORA #%01000000
0713- 8D 2C 06  0185 NONZERO    STA CURCHAR
                0186 ;
0716- A5 96     0187            LDA *ST       ;END OF TEXT ?
0718- F0 06     0188            BEQ OK
                0189 ;
071A- 20 CC FF  0190            JSR CLRCHN    ;RESTORE NORMAL I/O DEVS.
071D- A2 F8     0191            LDX #$F8      ;RESTORE STACK AND
071F- 9A        0192            TXS  ;GO BACK TO BASIC
                0193 ;
0720- 60        0194 OK         RTS
                0195 ;
0721- AE 28 06  0196 GETREST    LDX LINLEN    ;IGNORE REST OF LINE
0724- CA        0197            DEX
0725- 86 B4     0198 LOOP2      STX *SAVX     ;KEEP R(X)
0727- 20 06 07  0199            JSR GET
072A- A6 B4     0200            LDX *SAVX
072C- CA        0201            DEX
072D- EC 27 06  0202            CPX CHARTOT
0730- B0 F3     0203            BCS LOOP2
                0204 ;
0732- 60        0205            RTS
                0206 ;
                0207            .EJ
                0208            .EN
```

LABEL FILE: [/ = EXTERNAL]

```
/CHANNEL=0002          /ENDLIN=001F           /BLANK=0020
/LENGTH=0028           /DASH=002D             /SHFSPC=0060
/FORSPC=006F           /FORCMD=007A           /ST=0096
/SAVX=00B4             /SCREEN=8000           /IMAGES=8400
/SETINP=FFC6           /CLRCHN=FFCC           /WRT=FFD2
/GETCHR=FFE4           WORDTOT=0624           LINETOT=0626
CHARTOT=0627           LINLEN=0628            LINFLG=0629
BLNKFLG=062A           WORDFLG=062B           CURCHAR=062C
LASTCHAR=062D          START=062E             LOOP=0632
SETLEN=0638            EIGHTY=0646            FORTY=0647
SETCHN=064A            LOADADR=064F           LINESTRT=0655
ADDLINE=0660           CHKLINE=0675           FORMAT=068F
FINISH=0692            NOTFORMAT=0698         DONELINE=06A4
MORE=06A7              LEADBLK=06AC           COUNT=06C2
NOTLEAD=06C5           CONTLIN=06CD           NOTDASH=06E2
WORDCOUNT=06EC         DASHCOUNT=06F1         NOTYET=06FE
GET=0706               NONZERO=0713           OK=0720
GETREST=0721           LOOP2=0725
//0000,0733,0733
```

For Management Only

* * * MAGIS-PLUS * * *

The Superior Management Tool and Micro Accounting Software

MAGIS-Plus is a fully integrated system and includes a General Ledger, Accounts Receivable, Accounts Payable, Payroll, Inventory—all on one program disk, requiring only one data disk.

MAGIS is presently being used in the floor covering industry, picture frame shops, florist shops, graphic art studios, furniture manufacturers, general merchandise stores, home decorating centers, computer centers, bakeries, auto parts stores and many other small business establishments.

Feature	Benefit
A Sales Journal (cash and credit)	updates all affected files including General Ledger, prints invoices
An Expense System (cash and credit)	updates all affected files in General Ledger and subsidiary ledgers
An Accounts Receivable System	generates statements, produces aging and delinquency reports, records payments to affected files
An Accounts Payable System	records payment of bills, prints checks, tracks delinquencies, handles anticipated purchase discounts
A Perpetual Inventory System	produces physical inventory sheet and reorder reports
An Inventory Transaction System	records purchase orders and merchandise received and updates all affected accounts
A Payroll System	computes payroll, prints checks, produces W-2's and quarterly reports
A General Ledger System	produces trial balance, financial statements, comparative statements, and accommodates direct General Ledger postings

MAG™

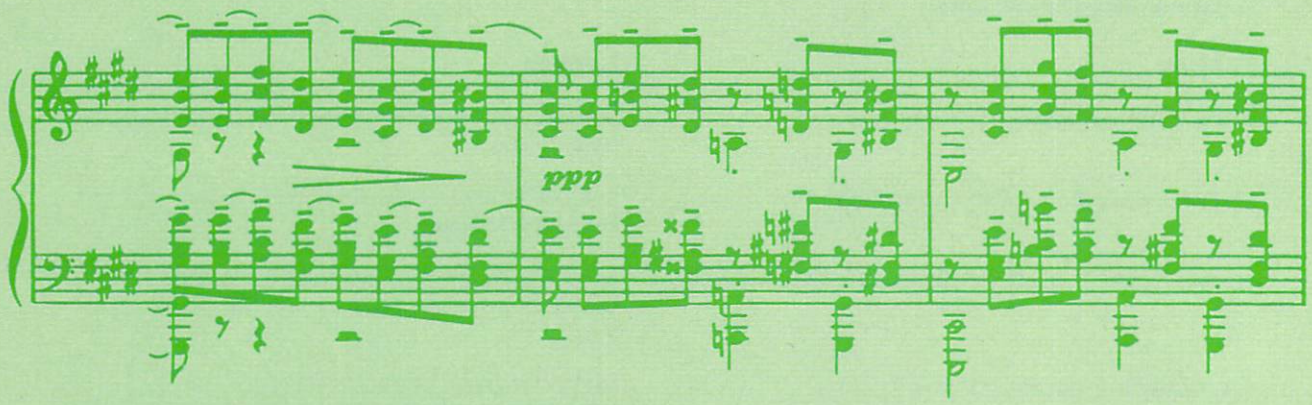
Merging computer technology with the practical worlds of business and government

OFFICIALLY APPROVED BY COMMODORE

The Management Accountability Group, Inc.
P. O. Box 346
Athens, GA 30603
(404) 353-8090

Studying Complex Musical Rhythms on the Commodore 64

PART ONE OF A SERIES OF
ARTICLES ON MAKING MUSIC
ON THE COMMODORE 64!
by LEE SILVAN



Computers as Musical Instruments

Throughout musical history the instruments with unique capabilities have endured, developed, and ultimately survived the tests of time and fashion. In many cases the uniqueness of an instrument is its sound quality or "timbre" (harp, oboe, flute, trumpet, and others), and in other cases it has been such things as range (contrabassoon, piccolo), or dynamic power (pipe organ). The complexity of rhythm in music has always been limited by the mental ability and coordination of the players. This is the main reason that rhythm as a musical element has not yet reached a state of complexity on a par with other elements like harmony and melody. The use of a computer to help develop fascinating, and otherwise unplayable rhythms is one of its unique capabilities as a musical instrument, and one which will help it gain acceptance as a bona fide musical instrument.

This series of articles is intended to whet the appetite of all the computerists who also happen to be musicians and composers. It is for those who share an interest in the use of computers in the arts as well as business and technology. The Commodore 64 is an ideal microcomputer for use in such a study since it is compact, portable, powerful, inexpensive, and contains a built-in four-channel sound synthesizer of immense and virtually untapped musical potential. Because of the power of the SID chip and the ease of programming it, the "64" will serve admirably in these studies.

Traditional Durational Symbols

It should be pointed out that in our system of musical notation we can see an amusing similarity between our rhythmic note values and the order of binary arithmetic:

1	2	4	8	16	32
WHOLE	HALF	QUARTER	EIGHTH	SIXTEENTH	THIRTY-SECOND

We will begin with a look at the very simple durational relationships that we use in traditional music and the symbols which represent these durations. This system of durations is based on symbols which represent an increase or decrease by a factor of two. Here is a table of common note values and their relationships.

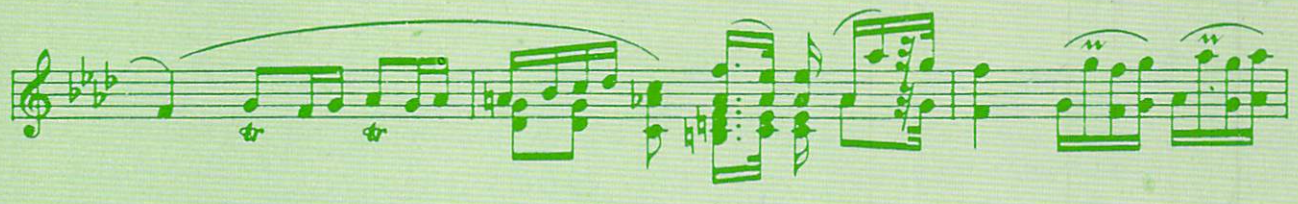
The WHOLE NOTE

The HALF NOTE (2 Halves = 1 Whole!)

The QUARTER NOTE (2 Quarters = 1 Half!)

The EIGHTH NOTE (2 Eighths = 1 Quarter!)

The SIXTEENTH NOTE (2 Sixteenths = 1 Eighth!)



Since each note is smaller by a factor of fifty-percent, it would seem that relationships by a factor of three would be a great deal more difficult, or possibly necessitate new note symbols. This is not so! With the addition of a dot (.) after a note value, there is an increase in its length by 50%.

A DOTTED WHOLE NOTE equals 3 or 2

A DOTTED HALF NOTE equals 3 or 2

A DOTTED QUARTER NOTE equals 3 or 2

A DOTTED EIGHTH NOTE equals 3 or 2

A DOTTED SIXTEENTH NOTE equals 3 or 2

Rests

There are times when it is necessary to show a specific amount of silence in a piece of music. In such cases musicians use symbols called rests. You may think of a rest as a silent note. Rests may also be dotted just like their note equivalents, and are named after them. Here is a list of the most common rests and their durational equivalents.

REST	NOTE
WHOLE	
HALF	
QUARTER	
EIGHTH	
SIXTEENTH	

Triplets

Sometimes it is necessary to divide an undotted note into three equal parts. This is shown in musical notation by a triplet symbol. In most cases this is indicated by a small '3' over a note grouping of the next smaller value than the note being divided. Here are the most common examples:

A half note ( ) divided into three parts is

A quarter note ( ) divided into three parts is

An eighth note ( ) divided into three parts is

Since it is the relative durations of notes which we perceive collectively as rhythm, and since we will determine these durations in our programming by assigning a number to each sound which is directly proportional to its length, we must choose these numbers with some care. In some cases a nice round number value like 250 might seem a good choice for a quarter note value. There are some pitfalls however.

If you know that a specific note value will be divided into three equal parts in some places, and two equal parts in other places, you must choose a number for that note that is equally divisable by both 3 AND 2! Choosing 250 for a quarter note value in such a case will not work since 250 is not equally divisable by both 3 AND 2! A better choice is 252. The value of this choice is shown by the table below. Not only will quarter notes be divisable by three OR two, but so will the smaller and larger note values. This is important to remember!

NOTE TYPE	NOTE VALUE	DIVIDED BY 2	DIVIDED BY 3
	126	63	42
	252	126	84
	504	252	168
	1008	504	252

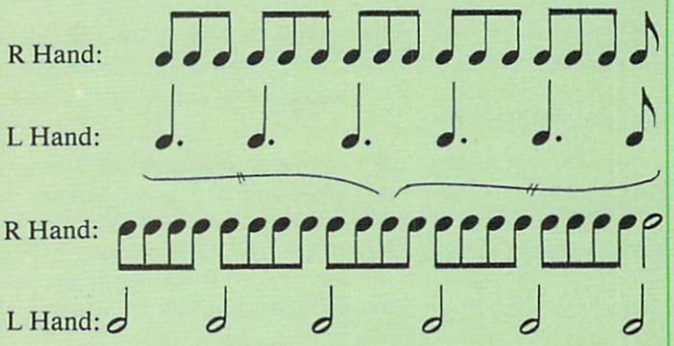
Easy Rhythms

Performing musicians subdivide everything according to portions of a "beat" which is constantly "ticking" in their head. Based on this metronomic sense of how long a beat is they can easily subdivide by 2, 3, 4, 6, or 8 depending on tempo. Good musicians with finely developed rhythmic sense can also subdivide a beat into 5 or 7 equal parts. Here are some easy rhythms for you to try. The first requires that you tap twice with your right hand to each tap of the left hand. in other words, divide the left hand beats into two EQUAL parts. The music looks like this:

R Hand:

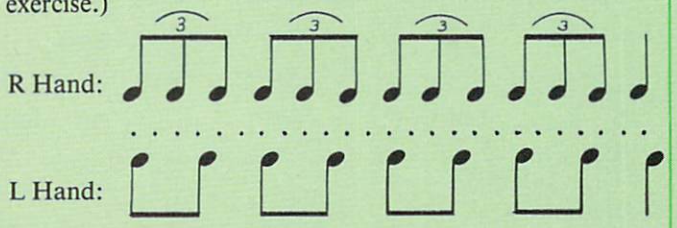
L Hand:

With minimum practice you should be able to do the following as well. First, three notes against one, then four against one.



Hopefully you have also felt some of the evocative power in these simple rhythms. Three beats against one reminds us of the long honored waltz meter. Four beats against one may remind you of the sound of the drums in much music by the American Indian. These rhythmic "styles" developed because musicians had the ability to subdivide large notes into specific groups of smaller notes.

You will develop a healthy respect for good percussionists when you try to divide three beats equally against two. Here is what it looks like in music notation. (I have added a line of evenly spaced dots to aid your accuracy in this exercise.)



Other Rhythmic Combinations

After struggling with this simple relationship of subdividing three evenly against two you will understand why not much music has been written in which eleven beats have been divided evenly against seven. That doesn't mean that the result would not be interesting musically. By choosing the proper poke values for duration we can let our Commodore 64 show us what this, or any other unconventional rhythmic combination would sound like.

Although most competent musicians can perform combinations of double and triple divisions of the beat, the other ratios are either much harder or virtually impossible—for us humans! We simply do not have the mental ability and coordination to perform combinations such as eleven notes divided evenly against five. Your Commodore 64 can provide you with the capability to hear and experiment with such fascinating rhythmic combinations, and quite easily!

Based on the approach already discussed on dividing three notes evenly against two, we can readily discover how to divide four notes against five. This is a musical capability inherent in the computer when used as a musical instrument.

If we wish to hear what the rhythmic combination of four notes divided evenly against five sounds like we simply need to choose an appropriate number (evenly divisable by both 4 & 5), and subdivide it appropriately. We find our least common denominator by multiplying the two numbers which constitute our ratio, in this case 4 and 5. Their product is twenty and therefore any number that is a direct multiple of twenty will do nicely. The smaller the number, the faster the rhythm! In this case let's use 260 for the length value of the longer note. (The one we want four of!) The length value for the shorter note will be (4*260)/5 or (1040)/5 which is 208. So, our two poke values for this particular rhythmic combination will be 260 and 208.

```
260   +   260   +   260   +   260   =   1040
!         !         !         !         !
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
                                        (each dot = 26!)
!         !         !         !         !
208   +   208   +   208   + +208   +   208   = 1040
```

Since the sound chip in our Commodore 64 allows us up to four simultaneous channels of sound, we can easily produce amazingly complex rhythmic patterns in our music such as 4:5:7:. Exciting, isn't it! Next time we'll have a program that allows you to do just that! C=

The Computer Becomes a Synthesizer

by Kent A. Multer

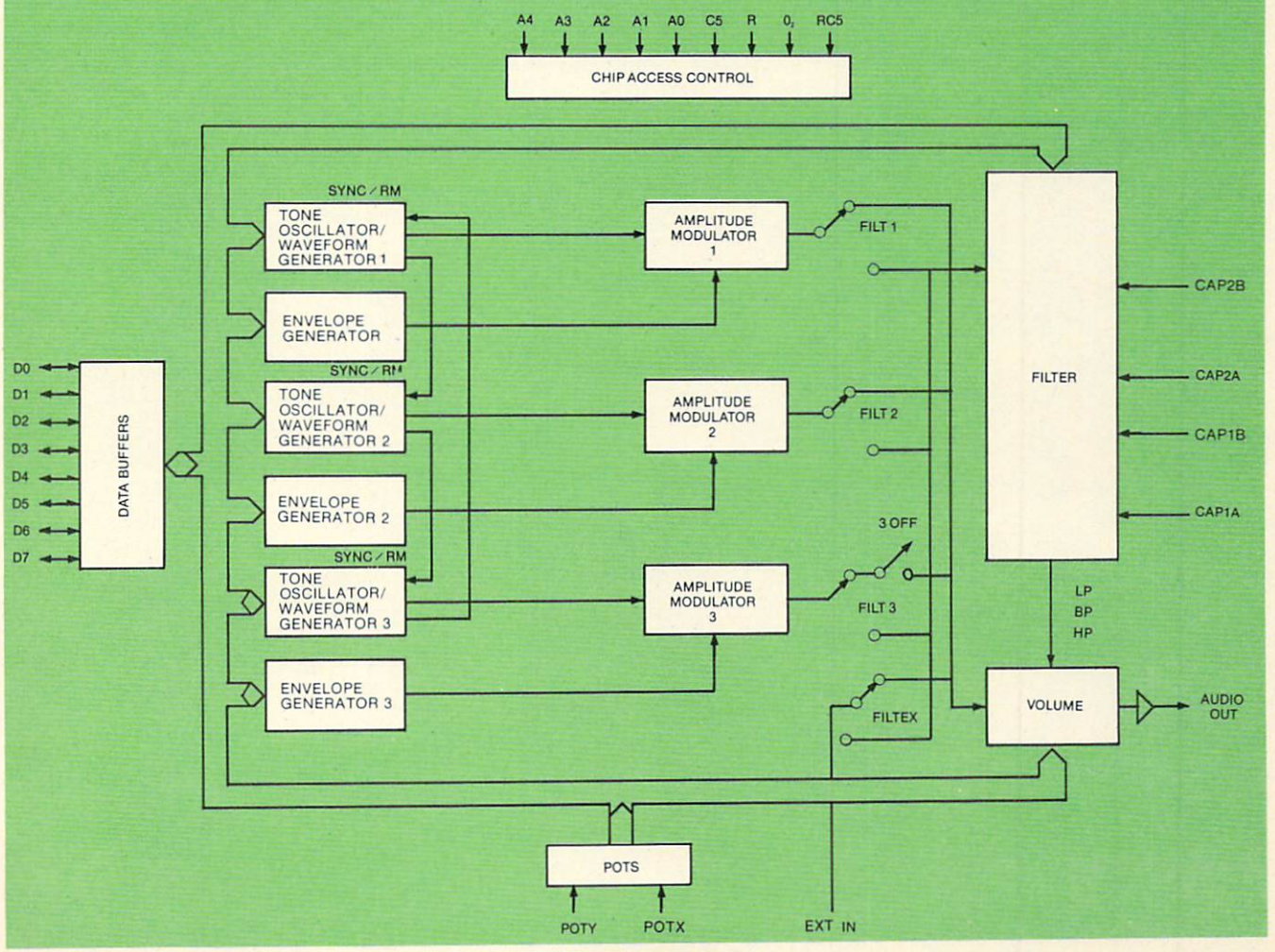
One of this year's most exciting new pieces of computer hardware is Commodore's model 6581 Sound Interface Device (SID). It's truly impressive: an entire music synthesizer on a single chip. SID is standard equipment in Commodore's newest computers, including the model 64, an advanced home computer with 64K of RAM. In this article I will describe SID's features, and give you some ideas on how it can be used.

Overview

Figure 1 shows a block diagram of the chip. As you can see, it has three voices, meaning it can produce three notes at once. Each voice consists of a tone generator, which produces the sound, and an envelope generator, which controls the volume. There are also some modulation effects, in which two tone generators combine to produce one complex sound.

The signals from the voices may be routed through a filter, which acts like a super tone control. This is the thing that makes SID so powerful, and so much more versatile

Figure 1 *Block diagram of the Commodore 6581 Sound Interface Device (SID).*



than the sound generators on other home computers.

Other features of SID include a master volume control, and an external audio input that allows you to run a signal from your electric guitar or other source through SID's filter. There are also two A/D converters on the chip, intended for connecting to pots. These are not electrically connected to anything else in SID, so you can use them for game controllers or whatever.

SID's functions are controlled by a number of 8-bit registers. In Commodore's machines, the chip is mapped into the memory address space, so you can write data into the registers with POKE statements. Each voice has seven registers for controlling its specific functions, and there are 8 more registers for controlling the filter, master volume, etc. Figure 2 illustrates the control registers.

Tone Generators

SID has three tone generators, each of which can generate four different waveforms: sawtooth, triangle, pulse, and noise. Sawtooth waves contain lots of harmonics, and are good for rich sounds like horns or strings. Triangle waves only have a little bit of harmonic energy, so they have a very mellow, flute-like sound.

Pulse waves can have a lot of different sounds, depending on the pulse width, which you can vary. When the pulse width is at or near 50%, you set a square wave, which is sort of hollow or clarinet-like. When the pulse width is close to zero (or to 100%; they both sound the same to the ear), the tone is very thin, more like an oboe. In between is an impressive variety of sounds, including saxophones, and sometimes human voices, if you set the filter right.

Figure **2** *SID control registers.*

Address	‡)	DATA D7	D6	D5	D4	D3	D2	D1	D0	REG NAME
										VOICE 1
54272		F_7	F_6	F_5	F_4	F_3	F_2	F_1	F_0	FREQ LO
54273		F_{15}	F_{14}	F_{13}	F_{12}	F_{11}	F_{10}	F_9	F_8	FREQ HI
54274		PW_7	PW_6	PW_5	PW_4	PW_3	PW_2	PW_1	PW_0	PW LO
54275		—	—	—	—	PW_{11}	PW_{10}	PW_9	PW_8	PW HI
54276		NOISE				TEST	RING MOD	SYNC	GATE	CONTROL REG
54277		ATK_3	ATK_2	ATK_1	ATK_0	DCY_3	DCY_2	DCY_1	DCY_0	ATTACK / DECAY
54278		STN_3	STN_2	STN_1	STN_0	RLS_3	RLS_2	RLS_1	RLS_0	SUSTAIN / RELEASE
										VOICE 2
54279		F_7	F_6	F_5	F_4	F_3	F_2	F_1	F_0	FREQ LO
54280		F_{15}	F_{14}	F_{13}	F_{12}	F_{11}	F_{10}	F_9	F_8	FREQ HI
54281		PW_7	PW_6	PW_5	PW_4	PW_3	PW_2	PW_1	PW_0	PW LO
54282		—	—	—	—	PW_{11}	PW_{10}	PW_9	PW_8	PW HI
54283		NOISE				TEST	RING MOD	SYNC	GATE	CONTROL REG
54284		ATK_3	ATK_2	ATK_1	ATK_0	DCY_3	DCY_2	DCY_1	DCY_0	ATTACK / DECAY
54285		STN_3	STN_2	STN_1	STN_0	RLS_3	RLS_2	RLS_1	RLS_0	SUSTAIN / RELEASE
										VOICE 3
54286		F_7	F_6	F_5	F_4	F_3	F_2	F_1	F_0	FREQ LO
54287		F_{15}	F_{14}	F_{13}	F_{12}	F_{11}	F_{10}	F_9	F_8	FREQ HI
54288		PW_7	PW_6	PW_5	PW_4	PW_3	PW_2	PW_1	PW_0	PW LO
54289		—	—	—	—	PW_{11}	PW_{10}	PW_9	PW_8	PW HI
54290		NOISE				TEST	RING MOD	SYNC	GATE	CONTROL REG
54291		ATK_3	ATK_2	ATK_1	ATK_0	DCY_3	DCY_2	DCY_1	DCY_0	ATTACK / DECAY
54292		STN_3	STN_2	STN_1	STN_0	RLS_3	RLS_2	RLS_1	RLS_0	SUSTAIN / RELEASE
										FILTER
54293		—	—	—	—	—	FC_2	FC_1	FC_0	FC LO
54294		FC_{10}	FC_9	FC_8	FC_7	FC_6	FC_5	FC_4	FC_3	FC HI
54295		RES_3	RES_2	RES_1	RES_0	FILT EX	FILT 3	FILT 2	FILT 1	RES / FILT
54296		3 OFF	HP	BP	LP	VOL_3	VOL_2	VOL_1	VOL_0	MODE / VOL
										MISC
54297		PX_7	PX_6	PX_5	PX_4	PX_3	PX_2	PX_1	PX_0	POTX
54298		PY_7	PY_6	PY_5	PY_4	PY_3	PY_2	PY_1	PY_0	POTY
54299		O_7	O_6	O_5	O_4	O_3	O_2	O_1	O_0	OSC3 / RANDOM
54300		E_7	E_6	E_5	E_4	E_3	E_2	E_1	E_0	ENV3

The noise waveform has no precise pitch; it's used for untuned sounds like percussion, wind, or jet engines. With this waveform, low notes come out as a deep rumble, and high notes sound like a snake's hiss.

The frequency of each tone generator is set by a 16-bit number that you POKE into two control registers (high and low bytes). The output frequency in Hertz is equal to the number in the registers multiplied by 0.0596. Table 1 gives the numbers that you use for the notes of the musical scale. SID has a range of eight octaves; is that enough for you? It ought to be; it's more than almost any conventional instrument can play.

With 16-bit frequency control, there are a lot of pitches in between the notes of the scale. You can generate glissando or portamento effects by rapidly incrementing or decrementing the number in these registers, so that the sound makes a smooth sweep from one pitch to another. You can also set two or three tone generators to be just a tiny bit out of tune with each other, which gives a rich, chorus-like quality to the sound.

Actually, SID's range is more than eight octaves: it can be tuned so low that you can't hear it, down to about 1 cycle every 16 seconds! We'll look at uses for the sub-audio range a little later.

Each voice has a control register that contains one bit for each of the four waveforms. If you turn on more than one of these bits, the resulting sound will be a logical ANDing of the selected waveforms. This could give you some interesting effects, but usually you will only use one waveform at a time. A word of warning: combining the noise waveform with any others may "lock it up", cancelling the noise output until you reset it with the Test bit or the chip Reset line.

When you select the pulse wave, the pulse width is set by a 12-bit number, which occupies two control registers. You can smoothly sweep the pulse width from one value to another, which gives a very nice soaring or "phase shifting" sound.

The control register contains bits for ring modulation and synchronization functions. These two effects are similar: they both take input from two tone generators, and produce an output that has some components of the inputs, plus some other frequencies that aren't present in either input. This can produce metallic sounds such as chimes and gongs. If you vary the frequency of one of the inputs while listening to the signal, you get a great science fiction-type sound in which you can hear some pitches rising, while others are falling at the same time. Note that the ring-mod function only affects the triangle wave output, but the synch function applies to all waveforms.

The difference between the two effects is something that I can't really describe in words, so I suggest you just try it. For some really wild sounds, you can use both effects at once. For example, you can set voice 2 to be in synch with voice 1, and set voice 3 to be ring-modulated by voice 2. I once did something like that with an ARP 2600 synthesizer, and got a really nice simulation of someone banging on a garbage can.

Each voice has a Test bit that, when set to 1, turns off all waveforms and resets the internal counters to zero. Commodore's spec sheets suggest that this feature may have some musical applications, but do not give any specific uses. I do have a couple of ideas, though: if you want to create a complex sound by combining two or three voices, you can use the Test bit to make sure that all the tone generators start their waveforms at the same moment; otherwise the slight delay might produce random variations in the sound. Also, the Test bit can be used to turn a voice on and off instantly, whereas using the envelope generator takes at least a few milliseconds.

Envelope Generators

This is another important synthesizer function. The term "envelope" refers to the way in which the volume changes during the playing of a note. Each note is divided into four phases called attack, decay, sustain, and release. In the attack phase, the volume rises from zero to a maximum or peak value. Then, during the decay, the volume falls off to some intermediate level. Next comes the sustain, in which the volume remains constant for as long as you want to hold the note. Finally, during the release, the volume falls back to zero.

The attack, decay, and release times, as well as the difference between the peak and sustain volume, are important factors in making one instrument sound different from another. For instance, as Figure 3 shows, a trumpet has a very short attack and decay time, giving a quick snap of loud sound at the beginning of each note. Then the volume remains constant as long as the musician keeps blowing, and when he or she stops, the note takes a tenth of a second or so to die out (release). Compare this to the violin envelope, which has a slow attack and no pronounced peak. The xylophone, by contrast, has a very fast attack, but no sustain at all; the note always dies away quickly.

Each voice in SID has its own controls for attack, decay, sustain and release. Each of these parameters is controlled by a four-bit number that can select one of sixteen possible values. The attack and decay are set by one control register, and the sustain and release by another. The attack times range from 2 msec. to 8 seconds. The decay and release times range from 6 msec. to 24 seconds. The sustain is not a time; it's a volume. If it is set to maximum (15), the volume will remain at the peak level, like the violin envelope in Figure 3.

The envelope generator is activated by a bit in the voice's control register called the Gate. (This is a synthesizer term, not really related to the logic gates we computer hackers are used to.) Setting the Gate to 1 starts a note; it causes the envelope generator to do its attack-decay-sustain phases. When the Gate is set to 0, the note begins its release phase. Note that the attack has a linear slope, but the decay and release have an exponential curve. This is a nice touch; it corresponds to the way that strings, ▶

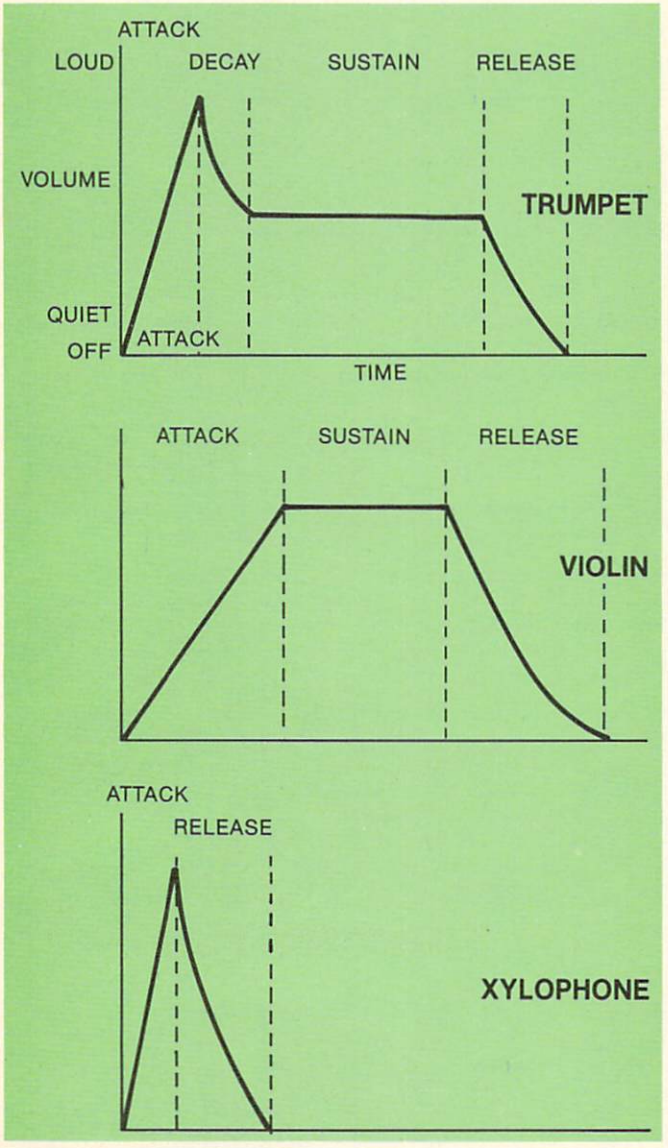
TABLE 1

Music Note Values

This contains a complete list of Note numbers, actual note, and the values to be POKEd into the HI FREQ and LOW FREQ registers of the sound chip to produce the indicated note.

Note	Note–Octave	Hi Freq	Low Freq
0	C–0	1	18
1	C#–0	1	35
2	D–0	1	52
3	D#–0	1	70
4	E–0	1	90
5	F–0	1	110
6	F#–0	1	132
7	G–0	1	155
8	G#–0	1	179
9	A–0	1	205
10	A#–0	1	233
11	B–0	2	6
12	C–1	2	37
13	C#–1	2	69
14	D–1	2	104
15	D#–1	2	140
16	E–1	2	179
17	F–1	2	220
18	F#–1	3	8
19	G–1	3	54
20	G#–1	3	103
21	A–1	3	155
22	A#–1	3	210
23	B–1	4	12
24	C–2	4	73
25	C#–2	4	139
26	D–2	4	208
27	D#–2	5	25
28	E–2	5	103
29	F–2	5	185
30	F#–2	6	16
31	G–2	6	108
32	G#–2	6	206
33	A–2	7	53
34	A#–2	7	163
35	B–2	8	23
36	C–3	8	147
37	C#–3	9	21
38	D–3	9	159
39	D#–3	10	60
40	E–3	10	205
41	F–3	11	114
42	F#–3	12	32
43	G–3	12	216
44	G#–3	13	156
45	A–3	14	107
46	A#–3	15	70
47	B–3	16	47
48	C–4	17	37
49	C#–4	18	42
50	D–4	19	63
51	D#–4	20	100
52	E–4	21	154
53	F–4	22	227
54	F#–4	24	63
55	G–4	25	177
56	G#–4	27	56
57	A–4	28	214
58	A#–4	30	141
59	B–4	32	94
60	C–5	34	75
61	C#–5	36	85
62	D–5	38	126
63	D#–5	40	200
64	E–5	43	52
65	F–5	45	198
66	F#–5	48	127
67	G–5	51	97
68	G#–5	54	111
69	A–5	57	172
70	A#–5	61	126
71	B–5	64	188
72	C–6	68	149
73	C#–6	72	169
74	D–6	76	252
75	D#–6	81	161
76	E–6	86	105
77	F6	91	140
78	F#–6	96	254
79	G–6	102	194
80	G#–6	108	223
81	A–6	115	88
82	A#–6	122	52
83	B–6	129	120
84	C–7	137	43
85	C#–7	145	83
86	D–7	153	247
87	D#–7	163	31
88	E–7	172	210
89	F–7	183	25
90	F#–7	193	252
91	G–7	205	133
92	G#–7	217	189
93	A–7	230	176
94	A#–7	244	103

Figure 3 *Sound envelopes of some typical instruments.*



horns, and other vibrating objects generally behave. Most synthesizers do either linear or exponential slopes, but not both.

Filter Section

The filter, in a sense, is the heart of a synthesizer. Granted, it doesn't actually produce sound, it just modifies what the tone generators produce. However, you will find that the filter has more control than anything else over what you hear. I'd rather have one waveform and a filter than a whole lot of waveforms and no filter.

The filter's function is similar to the tone controls on a hi-fi, in that it allows you to emphasize or remove certain parts of the audio spectrum. The range of possible effects is shown graphically in Figure 4.

SID's filter has three outputs. The low-pass output, as its name implies, will pass all signals below a certain frequency, called the cutoff frequency. Everything above the cutoff is "rolled off" (reduced in volume) at a rate of 12 dB per octave; the higher the frequency, the more it is reduced. Similarly, the high-pass output passes signals above the cutoff, and rolls off everything below it. The band-pass output rolls off frequencies above or below, and passes only those signals that are fairly close to the cutoff frequency.

Each of the filter's outputs is controlled by a single bit in one of SID's registers, so by setting several bits to 1, you can mix the outputs. Mixing the high-pass and low-pass outputs gives what is called a notch filter. This is the opposite of the band-pass: it rejects frequencies near the cutoff, and passes everything else.

The cutoff frequency is selected by an 11-bit number that you poke into two control registers. With the recommended 2200pf capacitors, the cutoff frequency can range from 30Hz to 10kHz.

The filter also has a resonance or "Q" control. This determines how strong the effect is. With low resonance, the sound is not too different from your hi-fi's tone controls. With high resonance, you get a very intense effect, like different vowel sounds of the human voice. A rock guitarist's wah-wah pedal is just a band-pass filter with a very high resonance. It goes "wah" when the musician raises the cutoff frequency by pushing down on the pedal, and it goes "yow" when he or she lowers the cutoff by moving the pedal the other way. You can produce the same effect with SID by selecting a high resonance, and varying the cutoff while a note is playing.

Most synthesizers provide an envelope generator that controls the filter cutoff, allowing it to automatically rise and fall every time a note is played. SID does not provide this feature, but you can write a program to do it. Some of the more expensive synthesizers can also change the resonance in the same manner. The effect is more subtle, but useful to the advanced synthesist. With SID, once again, you can do this under program control; isn't software wonderful?

Each of the three voices can be routed through the filter, or it can be sent directly to the main output. This feature helps make up for the fact that there are three voices but only one filter. You can filter one voice, and use pulse width changes to produce filter-type effects on the other two.

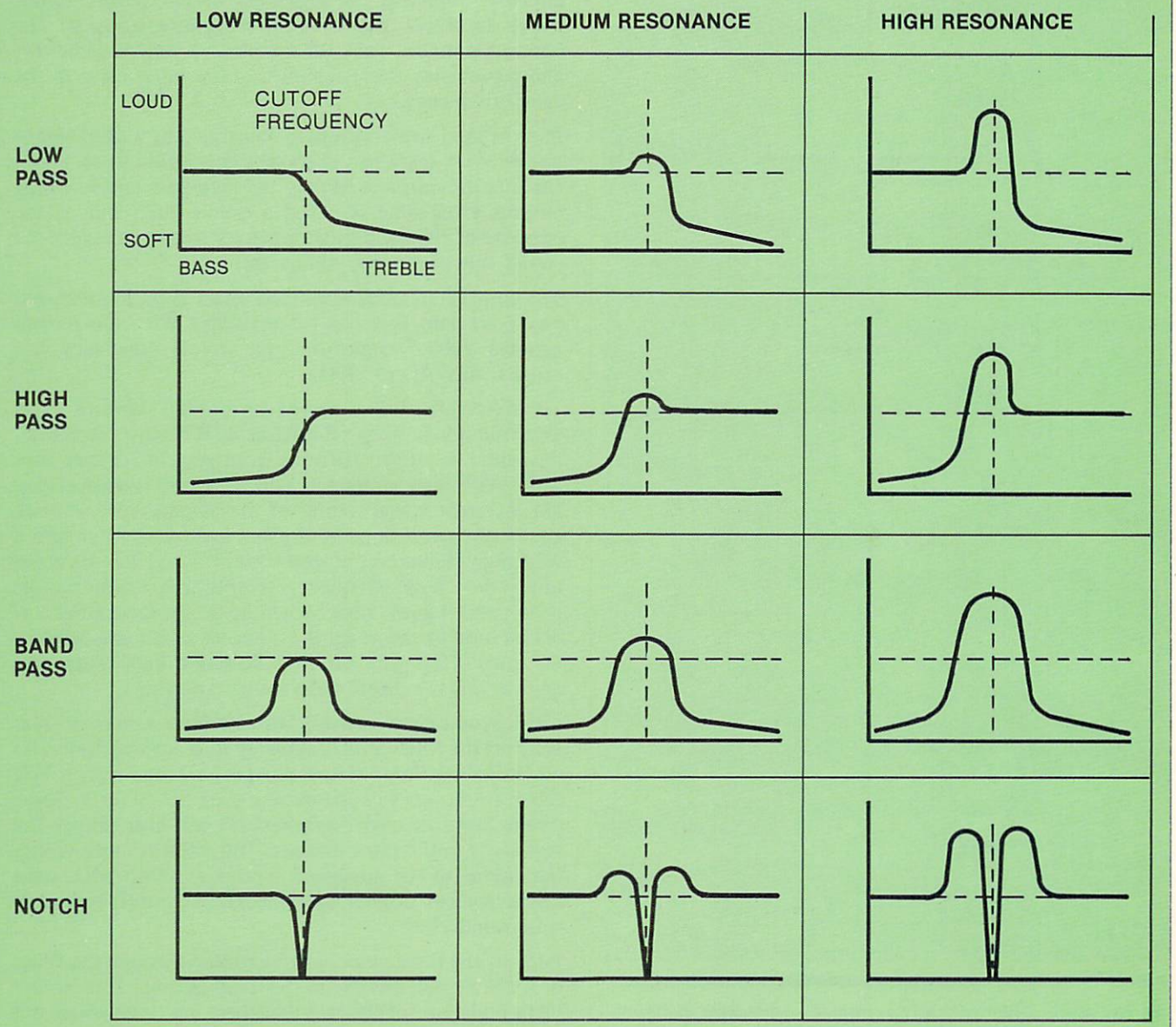
Other Features

As I mentioned earlier, SID has an external input that can be used with any instrument, recording, or even a microphone. This input can be sent through the filter, or it can go directly to the output. With a microphone or instrument, you may need to amplify the signal before running it into SID. The maximum input is 3 volts peak-to-peak.

SID has a master volume control that ranges from zero to 15. This controls all the voices, the filter, and the external input. You can produce tremolo effects by rapidly raising and lowering the volume.

Voice 3 has a couple of special features that are quite handy. There is a register that you can read to get the ▶

Figure 4 *Some effects that SID's filter can produce.*



instantaneous value of the tone generator's output, and another that lets you read the envelope generator's output. What are these good for? Well, earlier I mentioned that it would be nice to have another envelope generator to control the filter. You can use voice 3's envelope generator to do this, by putting some statements in your program to read the envelope value, and POKE it into the filter cutoff.

You can use the waveform value register in a similar manner. For instance, set up voice 3 to produce a triangle wave at a very low frequency, say 3 or 4 Hz. Then write a program that continuously reads the waveform value, and adds it to the frequency numbers for voice 1 and/or 2. The result is vibrato: continuous up-down variation of the pitch.

Of course, when you're using voice 3 in this manner, you probably won't want to listen to it at the same time. Fortunately, there is a control bit that disables voice 3's output, so that it won't produce unwanted sounds.

One other use for voice 3: if you set it to the noise waveform at some very high frequency, the waveform value at any instant is essentially random. Thus you have an instant random number generator.

SID contains two A/D converters that are intended for connection to pots, for use as game paddles or other control functions. You might want to go to a music store,

and buy an old used wah-wah pedal. Tear out all the electronics, put in a 470K pot, and run it to one of the A/D inputs. Presto, you've got a wah-wah pedal for your computer . . . or a volume pedal, or a tremolo pedal, etc. Like I said, isn't software wonderful?

The A/D inputs are scanned about once every half a millisecond, so it's possible that they could be used to listen to an external sound and compute its pitch. Then SID could play or sing along with you.

Technical Details

Originally I was going to title this section "Bugs" or "Criticisms", but really, this chip is so wonderful that I couldn't bring myself to use such derogatory language. However, there are one or two things that I just couldn't resist pointing out.

Most of the control registers are "write-only," meaning that you can POKE things into them, but you can't PEEK at them to find out what their current settings are. So you may need to store their values in a table, and have your program update the table whenever it changes one of the registers.

If you've used some other synthesizers, you may notice that SID's filter effects are not quite as intense as you expected. This is because SID's filter has a 12 dB per octave roll-off, whereas most synthesizers have 24 dB per octave. However, you can always set two SIDs, and run one into the other through the external input. It'll still cost a heck of a lot less than a new Korg or MiniMoog.

Although SID has a master volume control, it does not have controls that let you set one voice louder or quieter than the others. This could be a problem if you're trying to play a fairly complex piece of music with a lot of variety between the voices. In a pinch, you could write a program to rapidly switch a voice's Gate signal on and off, so that the volume hovered around some intermediate level.

In summary, my hat is off to Commodore for producing a chip that turns a home computer into a real synthesizer. The musical world is bound to be rocked by this exciting new development. Does anyone out there want to buy a used Korg?

Kent Multer *(P. O. Box 732, W. Acton, MA 01720) wrote his first computer game in 1970, at the age of 14. He later attended Worcester (Mass.) Polytechnic Institute, where he did his first experiments with computer music. He is now a free-lance author and programmer.* C=

Microphys Programs

PET®

VIC-20™

2048 Ford Street
Brooklyn, New York 11229
(212) 646-0140

PET®

VIC-20™

FOREIGN AND DOMESTIC DISTRIBUTOR INQUIRIES WELCOMED.

Microphys, a leader in educational software development, is pleased to announce the release of several recreational software programs for use with the Commodore VIC-20 microcomputer. The VIC programs, described below, require a 3-K expansion cartridge and utilize the VIC's excellent color graphics and sound capabilities. Each program retails for $15 and is accompanied by complete instructions.

PROGRAM DESCRIPTIONS

PV901 - Missile Math: this program presents in a game format, an opportunity for youngsters (ages 5-15) to practice and develop the basic skills of addition, subtraction, multiplication, and division. Four levels of difficulty in each skill area may be selected. Problems in a given skill are randomly generated and missiles are launched at correct answers. The computer displays the results on each program run and may be directed to generate the same sequence of problems so that review and 'match play' (against an opponent) are possible.

PV350 - Cryptograms: this program permits the generation of 'secret' messages which are to be decoded. These cryptograms are displayed along with their unique code number classifications. To decode a cryptogram, the program is run from line 9000. Family members can challenge each other with their individually created messages. If you enjoy solving the cryptograms appearing in newspapers and crossword puzzle magazines, this program is perfect for you. Note: two VIC users may exchange encoded messages. User 1 creates a secret message and transmits this to user 2. The code number will permit user 2 to have **his** VIC decipher the message should he encounter any difficulty.

PV340-349 - Anagrams: this series of programs provides an educational challenge for virtually all age groups. The VIC randomly generates scrambled words which are to be identified. Two clues are provided in order to assist in this process. The clues in the school and college categories are generally definitive in nature. Many of the words used are part of the Microphys Spelling and Vocabulary series for the associated grade levels. Thus, reading, vocabulary, and spelling skills are reinforced by these Anagram programs. Note: the same sequence of words generated may be requested so that 'match play' is possible. There are 5 level-of-difficulty categories each consisting of two programs.

PV340-341 Recreational	PV342-343 College	PV344-345 High School
PV346-347 Junior High	PV348-349 Elementary	

PV375-380 - Wheel-of-Fortune Word Games: this series of programs represents an exciting challenge for every member of the family. Players try to fill in missing letters in a randomly generated title or phrase and earn and lose points according to the graphic display on a 'Wheel-of-Fortune'. The scores of as many as four players are displayed, 1000 points being required to win a given game.

PV375 Song Titles	PV376 Famous Places	PV377 Entertainers
PV378 Statesmen	PV379 Scientists	PV380 Sports Figures

PV601-644 - Missile Spelling: this series of 36 programs enables youngsters in grades 4 through 12 to practice and develop basic spelling skills. Each program contains 60 graded words. The VIC randomly selects groups of 5 words, one of which is spelled incorrectly. Missiles are launched in order to destroy the word misspelled. The words chosen for grades 7 - 12 correspond to the Microphys Vocabulary series. Note: there are 4 programs in each grade level.

PV601-604 Grade 12	PV606-609 Grade 11	PV611-614 Grade 10
PV616-619 Grade 9	PV621-624 Grade 8	PV626-629 Grade 7
PV631-634 Grade 6	PV636-639 Grade 5	PV641-644 Grade 4

PV401-460 - Vocabulary: each vocabulary program randomly generates graded words which are to be defined. A sentence, in which the word is properly used, is displayed when an incorrect response is made. Using this contextual clue, a second opportunity to define the word is given. Reading and spelling skills are also reinforced as a more powerful vocabulary is developed. There are 10 programs in each grade level.

PV401-405 and PV431-435 Grade 12	PV406-410 and PV436-440 Grade 11
PV411-415 and PV441-445 Grade 10	PV416-420 and PV446-450 Grade 9
PV421-425 and PV451-455 Grade 8	PV426-430 and PV456-460 Grade 7

Educators should write for the new Microphys Fall Catalog which describes over 200 programs for use in Chemistry, Physics, Calculus, Mathematics, Vocabulary, and Spelling classes on both the high school and college levels.

My Favorite VIC Cartridge

by Neil Harris

There is a question often asked of me. It's asked by many prople: friends, co-workers, business associates, and even total strangers. I'm known as something of a VIC expert, as well as a videogame maniac, and my word on software can reassure someone enough for them to make their choice.

"What is your favorite cartridge for the VIC 20?"

I have worked up a kind of ritual response to this. First I give a coy smile, keeping silent. This invariably results in the start of a guessing game.

Cosmic Cruncher? Powerfully appealing, and I'll admit spending countless hours at it, but it isn't the one.

Gorf? Omega Race? I've stayed awake late for these, too, honing my reflexes to hair-trigger quickness. Nonetheless, neither of these is my favorite one.

Then what about Garden Wars? Almost a psychedelic game, with more things happening at one time than seems reasonable; but, no, not this either.

Then surely it must be . . . Pinball Spectacular?

Here I pause dramatically, allowing an almost reverent silence to build, before admitting that this, too, was not my favorite.

At this point, the asker is puzzled. They name several additional popular games, but with less confidence now. Some of the trickier people ask if, perhaps, I consider the modem by favorite cartridge, or maybe the 16K memory expander.

At this point I shake my head gently, and move closer for a confidential whisper.

I tell them, "My favorite has the best graphics of any cartridge. If offers more variety to my VIC keyboard, along with better music."

Their eyes widen, but they still don't know which it is.

I finally tell. "The Super Expander, of course."

No one has guessed yet. I don't know why, but this seems to be the least understood of any cartridge that Commodore makes. The descriptions I've seen published dwell on the 3K of RAM it contains, and the fact that it allows programming the function keys to words, phrases, and program lines.

Perhaps it's the name. A more appropriate name was its original moniker, the Video Support Package, which wasn't

used because it didn't describe the other aspects of the cartridge, like music mode.

The Super Expander appeals to me because it makes easy some of the most popular and attractive features of the VIC: color graphics and music. And it performs its job in the best method possible, by melding them with the built-in BASIC in the computer.

Super Expander adds commands for graphics and color to the list of BASIC words that the VIC understands. When Super Expander is plugged into a VIC 20, programs can include these commands and draw high resolution lines, circles, and points on the screen, as well as the normal text. BASIC programs can easily paint in certain areas of the screen, switch colors and draw. The programmer can forget the long list of POKEs required for doing high resolution normally, because Super Expander handles them automatically.

Table 1: Super Expander Commands

Command	Purpose
GRAPHIC	Sets graphic mode
SCNCLR	Clears a graphic screen
COLOR	Set screen, border, character and aux. colors
REGION	Set character color only
DRAW	Plots a line
POINT	Plots a single point
CIRCLE	Plots a circle, ellipse or arc
PAINT	Colors in an enclosed area
CHAR	Puts text on graphic screen
SOUND	Sets 4 tones and volume
RGR	Read graphic mode
RCOLR	Read color register
RDOT	Read color of point
RPOT	Read position of game paddle
RPEN	Read position of light pen
RJOY	Read position of joystick
RSND	Read value in sound register

The music commands were incorporated into BASIC painlessly as well. Not through new commands, although there is a SOUND command that is useful for sound effects. Music mode works through the all-purpose PRINT command in normal BASIC.

The PRINT statement normally uses all sorts of control characters for cursor movement and color changes. Super Expander adds a new control character, accessed by holding down the CTRL key and pressing the left arrow (←), or by using the CHR$ code of 6. Once this character is PRINTed, the VIC is in music mode. All letters following the control character are interpreted as music commands. The letters A through G tell the computer to play the corresponding musical notes, and other letters and numbers set tempo, octave, and voice number.

Table 2: Music Mode Commands

Character	Purpose
C, D, E, F, G, A, B	Play note
#	Play next note sharp
$	Play next note flat
R	Rest for one beat
T	Set tempo
S	Set sound register (voice)
V	Set volume
O	Set octave
P	Print characters on screen
Q	Don't print characters on screen
RETURN	End music mode

I'm not musically inclined, really, but I've learned to play chords using simple PRINT statements. For example, the following program plays some simple chords:

Program 1: Simple Chords

```
10 PRINT CHR$(6); : REM ENTER MUSIC MODE
20 PRINT "T5 V9 S1 O3 C S2 O2 E S3 O1 G";
30 PRINT "S1 O3 D S2 O2 #F S3 O1 B" : REM END
   MUSIC MODE
```

I must admit that I don't spend much time playing with the music mode. That isn't my nature. What I do spend hours with is graphics.

Super Expander lets you thoroughly explore the VIC's graphic modes. The command GRAPHIC N sets up the screen in several different graphic modes.

GRAPHIC 2 sets the screen in high resolution. You can control a grid of 160 by 160 dots (which corresponds to 20 by 20 characters in text mode). The color resolution is limited in high resolution mode, which means that there can be only one color (plus the screen's color) in each eight-by-eight dot area (corresponding to one character in normal text mode). Any attempt to create a second color in an area will cause all dots in that area to change color.

GRAPHIC 1 is the multi-color mode. Now the screen grid is 80 by 160, with each dot twice the width of a high resolution dot. Each four-by-eight area can contain three colors (plus the screen's color). One of the colors is the character color, which has the same limitation as high resolution mode. The other two colors are the border and auxiliary colors.

Because the border and auxiliary colors can be used in dots on the screen, special effects can be created. At any time, when the border or auxiliary colors are changed, every dot on the screen which uses those colors is changed instantly. Dots

can appear, disappear and blink under the control of a simple BASIC program!

Super Expander is invaluable as an aid to graphic design and plotting. The manual that comes with the Super Expander cartridge contains three of my early experiments in programming with it, including a program that draws a picture of a kangaroo. Alongside this article you'll find two more programs, more recent attempts to create animated graphic displays.

So remember, the next time you're in your VIC dealer's store and looking for a sound investment in a cartridge, ask for the Super Expander. And tell them I sent you.

Program 2:

```
10 REM HYPNOTIC GRAPHICS --- VIC WITH SUPER
   EXPANDER
20 REM BY ---> NEIL HARRIS
30 DIM C(3)
40 FOR L=0 TO 3
50 GRAPHIC 1
60 FOR L=0 TO 3
70 C(L)= INT( RND(1)* 8* ((L/2 <> INT(L/2))
   +2))
80 NEXT
90 COLOR C(0), C(1), C(2), C(3)
100 XD= INT( RND(1)* 511)
110 YD= INT( RND(1)* 511)
120 C=INT( RND(1)* 4)
130 FOR L=0 TO π STEP π/ 60
140 X1= XC+ XD* COS(L)
150 Y1= YC+ YD* SIN(L)
160 X2= XC- XD* COS(L)
170 Y2= YC- YD* SIN(L)
180 DRAW C, X1, Y1 TO X2, Y2
190 GET A$
200 IF A$> "" THEN: GRAPHIC 4: END
210 NEXT: GOTO 60
```

Program 3:

```
10 REM KALEIDOSCOPE FOR VIC & SUPER EXPANDER
20 REM BY ---> NEIL HARRIS
30 GRAPHIC 1
40 DIM A(1),D(1),C(3),X(5),Y(5),X1(5),Y1(5)
50 DRAW 1,255,0 TO 0,511 TO 255,1023
60 DRAW 1,767,0 TO 1023,511 TO 767,1023
70 PAINT 1,0,0
80 PAINT 1,0,1023
90 PAINT 1,1023,0
100 PAINT 1,1023,1023
110 X= RND(-TI)
120 K= SQR(3)*256
130 K1= π/3: K2= π/6: K3=6/π : J=511
140 FOR L=0 TO 3
150 C(L)= INT( RND(1)*8*( ABS(L-1.5)+.5))
160 NEXT
170 FOR Q=1 TO 10
180 FOR L=0 TO 1
190 A(L)= RND(1)*K1
200 D(L)=J* RND(1)-(1- ABS(A(L)-K2)*K3)*69
210 NEXT
220 C= INT( RND(1)*4)
230 IF C>1 THEN C(C)= INT( RND(1)*8*(C-1))
240 COLOR C(0),C(1),C(2),C(3)
250 FOR L=0 TO 5
260 A=A(0)
270 IF L AND 1 THEN A=K1-A
280 X(L)=J+D(0)* COS(A+L*K1)
290 Y(L)=J+D(0)* SIN(A+L*K1)
300 A=A(1): IF L AND 1 THEN A=K1-A
310 Y1(L)=J+D(1)* SIN(A+L*K1)
320 X1(L)=J+D(1)* COS(A+L*K1)
330 NEXT
340 FOR L =0 TO 5
350 DRAW C,X(L),Y(L) TO X1(L),Y1(L)
360 NEXT
370 GET A$
380 IF A$>"" THEN: GRAPHIC4: END
390 NEXT: GOTO 140
```

High Resolution Screen Dump Program for the Commodore 64

by
Rick Sterling

Here is a "quick and dirty" BASIC program that will print the Commodore 64's high-resolution screen to the 1525 printer. The program assumes that your bit-map begins at 8192, but modifying line 10 can adjust this

```
5 OPEN 4, 4: DIM Z%(199)
10 S = 8192 : REM START OF BIT MAP
20 FOR I=39 TO 0 STEP -1: FOR J=0
   TO 24
30 FOR K=0 TO 7
40 Q = J* 8+ K
50 Z%(Q) = Z%(Q)+(PEEK( S+ 320* J+
   I* 8+ K)) *2↑Y
55 A$ = A$+ CHR$(( Z%(Q) AND 127)
   + 128)
60 Z%(Q) = Z%(Q) / 128
65 NEXT K: NEXT J
70 PRINT# 4, CHR$(8) A$: A$ = ""
80 Y = Y + 1: IF Y=7 THEN Y=0:
   GOTO 1000
100 NEXT I
999 R = 1
1000 FOR L=0 TO 199
1010 A$ = A$ + CHR$(( Z%(L) AND
     127) + 128)
1020 Z%(L) = Z%(L) / 128: NEXT L
1030 PRINT# 4, CHR$(8)A$: A$= ""
1040 IF R=0 THEN 100
1050 CLOSE 4
1060 END
```

3 PROGRAMS FOR THE 64

by Christopher J. Phillips

Currently, I'm primarily using my Commodore 64 as a terminal. The BASIC terminal program shown below is handy because I can load a program from it, enter CMD 5: LIST, and the program is uploaded (all upper case, unfortunately) to my computer at work. This must be followed by PRINT#5: CLOSE 5. Then I can reload the terminal routine and continue.

The mailing list program from the 1541 manual needs some help to be fully functional. Several slight errors are corrected as shown in the REM statement.

The HEXMON program lets you view the contents of memory in the 64. With the legendary *Programmer's Reference Guide* and a 6502 reference card, it will be really handy.

```
1 REM MAILING LIST ROUTINE FROM 1541 USER MANUAL
2 REM MODIFIED BY CHRIS PHILLIPS
3 REM CREATES NEW FILES, UPDATES MIDDLE RECORDS, ALLOWS NULLS,
4 REM CORRECTLY PROCESSES RECORD # 256
5 REM CHANGES TO LINES LINES 25, 28, 70, 110, 220, 245
9 A(1)=12: A(2)=15:A(3)=20:A(4)=20:A(5)=12:A(6)=2:A(7)=9:A(8)=10:X=1
10 OPEN1,8,15:OPEN2,8,3,"0:MAILING LIST,L,"+CHR$(108):GOSUB900
20 PRINT#1,"P"CHR$(3)CHR$(1)CHR$(0)CHR$(1)
25 GOSUB 900:IF E=50 THEN PRINT#2,1:GOTO 20:REM INITIALIZE
28 INPUT#2,X$ : IF VAL(X$)>0 THEN X=VAL(X$)
30 INPUT"READ, WRITE, OR END";J$:IFJ$="E"THENCLOSE2:CLOSE1:END
40 IF LEFT$(J$,1)="W"THEN200
50 PRINT:INPUT"RECORD#";R:IFR<0ORR>XTHENPRINT"RECORD NOT FOUND"
   :GOTO 50
60 IF R<2 THEN 30
70 R1=R:R2=0:IF R>255 THEN R2=INT(R1/256):R1=R1-256*R2
80 RESTORE:DATA1,FIRST NAME,14,LAST NAME,30,ADDRESS1,51,ADDRESS2
90 DATA 72,CITY,85,STATE,88,ZIP,98,PHONE#
100 FOR L=1TO8:READA,A$:PRINT#1,"P"CHR$(3)CHR$(R1)CHR$(R2)CHR$(A)
    :GOSUB900
110 ONA/99GOTO50:INPUT#2,Z$:PRINTA$;TAB(12);Z$:NEXT:GOTO50
200 PRINT:INPUT"RECORD #";R:IFR<0ORR>5000THEN PRINT"ILLEGAL RECORD"
    :GOTO 200
210 IF R<2 THEN 30
215 IF R>XTHENR=X+1:PRINT:PRINT"RECORD #";R
220 R1=R:R2=0:IFR1>255THEN R2=INT(R1/256):R1=R1-256*R2
230 RESTORE:FOR L=1TO8:READA,A$:PRINT#1,"p"CHR$(3)CHR$(R1)CHR$
    (R2)CHR$(A)
240 PRINTA$;:Z$=" ":INPUTZ$:IFLEN(Z$)>A(L)THENZ$=LEFT$(Z$,A(L))
245 PRINT#2,Z$:NEXT :IF R<X THEN 200
250 PRINT#1,"P"CHR$(3)CHR$(1)CHR$(0)CHR$(1): X=R
260 PRINT#2,X:GOTO200
900 INPUT#1,E,B$,C,D:IFE<20THEN RETURN
910 IF E<>50 THEN PRINT E;B$,C;D:STOP:RETURN
920 IF J$="R" THEN PRINT B$
930 RETURN
1000 OPEN 15,8,15:INPUT#15,E,ER$,B,S
1010 PRINT E;ER$;B;S:CLOSE15
```

```
1 REM TERMINAL WITH DISK, BY CHRIS PHILLIPS
2 REM VIRTUAL TERMINAL FOR 64
3 REM USE <CLR> FOR DISK COMMANDS, ? FOR HELP, SYS FOR TIME
4 REM ALL DISK COMMANDS EXCEPT LOAD (/PROGRAM) RETURN TO TERMINAL
5 REM LOAD ALSO LEAVES MODEM CHANNEL (#5) OPEN
10 REM
12 REM TO RESTART & SKIP INIT TYPE:  GOTO 210
100 OPEN 5,2,3,CHR$(6)
110 DIM F%(255),T%(255)
120 FOR J=32TO64:T%(J)=J:NEXT
130 T%(13)=13:T%(20)=8:RV=18:CT=0
140 FORJ=65TO 90:K=J+32:T%(J)=K:NEXT
150 FOR J=91 TO 95:T%(J)=J:NEXT
160 FOR J=193 TO 218:K=J-128:T%(J)=K:NEXT
170 T%(146)=16:T%(133)=16:T%(148)=127:T%(137)=17
175 T%(134)=03:T%(135)=01:T%(136)=18:T%(140)=20:T%(139)=15:T%(138)=19
180 FOR J=0TO255:K=T%(J)
185 K=T%(J)
190 IF K<>0 THEN F%(K)=J:F%(K+128)=J
200 NEXT
205 FOR I=1TO31:IF T%(I)=0 THEN T%(I)=I
206 NEXT I:REM TRANMIT CONTROL CHARS
209 REM CLR HOME,LOWER CASE,DISABLE CASE SHIFT
210 PRINT"  "CHR$(147)" "CHR$(14)" "CHR$(8)
220 PRINTTAB(6);"TERMINAL READY. ";TI$
310 GET#5,A$:REM ---MODEM INPUT---
320 IF A$="" OR ST<>0 THEN 360
330 PRINT " "CHR$(17)CHR$(F%(ASC(A$)));
335 IF ASC(A$)=10 THEN GOTO360
340 IF F%(ASC(A$))=34 THEN POKE 212,0
350 GOTO 310
360 PRINT CHR$(RV)" "CHR$(157);CHR$(146);
362 GET A$:REM ---KEYBOARD INPUT---
365 IF A$=CHR$(147)THEN GOSUB 400
370 IF A$<>"" THEN PRINT#5,CHR$(T%(ASC(A$)));
380 CT=CT+1:IF CT=8THENCT=0:RV=164-RV
395 GOTO 310
400 PRINT" ";:INPUT"WHAT";C$
410 IF LEFT$(C$,1)="@" THEN 500
420 IF LEFT$(C$,1)=">" THEN 600
430 IF LEFT$(C$,1)="/" THEN 700
440 IF C$="E"ORC$="END"ORC$="BYE"ORC$="QUIT"THEN PRINTCHR$(9),TI$:END
450 IF C$<>"HELP" AND C$<>"?" THEN 480
452 PRINT" ENTER:","'HELP' FOR THIS MESSAGE, OR"
455 PRINT,"'@' FOR DISK STATUS, OR":
    PRINT,"'>' FOR DISK COMMAND, OR"
460 PRINT,"'/NAME'  TO LOAD PROGRAM, OR":
    PRINT,"'END' TO EXIT":GOTO 490
480 PRINTCHR$(17)"                                                "CHR$(17)
490 A$="": C$="":RETURN
499 REM ---DISK STATUS DISPLAY---
500 OPEN 1,8,15:REM DISK CONTROL CHANNEL
510 INPUT#1,E,B$,C,D:PRINT E;B$;C;D
```

```
520 CLOSE1:GOTO490
599 REM --- DISK COMMAND ---
600 OPEN 1,8,15:REM DISK CONTROL CHANNEL
610 PRINT#1,RIGHT$(C$,LEN(C$)-1)
650 INPUT#1,E,B$,C,D:IF E>20 THEN PRINT E;B$;C;D
690 CLOSE 1: GOTO 490
699 REM --- LOAD COMMAND ---
700 LOAD "0:"+RIGHT$(C$,LEN(C$)-1),8
710 GOTO 650



1 REM HEXMON FOR 64 BY CHRIS PHILLIPS
2 REM TYPE H AND THE ADDRESS TO CONVERT TO DECIMAL
3 REM TYPE THE ADDRESS TO VIEW CONTENTS IN HEX
4 REM TYPE D AND THE ADDRESS TO VIEW IN DECIMAL
5 REM HIT RETURN FOR NEXT BLOCK
6 INPUT"DATE";DATE$
10 Y=16:Z=32
20 DEF FNMD(X)=INT(X-INT(INT(X/Y)*Y))
30 H$="0123456789ABCDEF": GOTO 220
40 INPUT "ADDRESS";A$
50 IF A$="END"ORA$="BYE"ORA$="E"THEN PRINT:PRINT,"OFF AT ";TI$:END
52 IF LEFT$(A$,1)="D" THEN 110
54 IF LEFT$(A$,1)="C" THEN 160
56 IF LEFT$(A$,1)="H" THEN 180
58 IF A$="SYS"THEN 220
60 A=VAL(A$):FOR I=ATOA+Z STEP 8:D=I
70 HX$="":GOSUB250:PRINTHX$;TAB(7)
80 FOR J=ITOI+7
90 PRINT MID$(H$,FNMD(PEEK(J)/16)+1,1);MID$(H$,FNMD(PEEK(J))+1,1);"  ";
100 NEXT J :PRINT: NEXT I :A$=STR$(I):GOTO 40
110 A=VAL(RIGHT$(A$,LEN(A$)-1))
120 FOR I=A TO A+ Z STEP 8
130 PRINT I;:L = 1:FOR J=ITO I+7 STEP 2
135 PRINT TAB(L*8);:L=L+1
140 PRINT PEEK(J)+PEEK(J+1)*256;:NEXT J
150 PRINT :NEXT I:A=I:A$="D"+STR$(A):GOTO 40
160 A=VAL(RIGHT$(A$,LEN(A$)-1))
170 FOR I=A TO A+199:PRINT CHR$(PEEK(I));:NEXT:A$="C"+STR$(I):GOTO 40
180 J=LEN(A$) :D=0:FOR I=2TOJ :K=1
190 IF MID$(H$,K,1)<>MID$(A$,I,1)THEN K=K+1:GOTO 190
200 D=D+(K-1)*16 ↑ (J-I) :NEXT I
210 PRINTCHR$(17)TAB(20);"=";D:A$=STR$(D):GOTO 40
220 PRINTCHR$(147):PRINT:PRINT TAB(8);
225 PRINT DATE$;SPC(3);LEFT$(TI$,2);":";MID$(TI$,3,2);":";RIGHT$(TI$,2)
230 PRINT:PRINT:GOTO 40
250 REM -- LONG HEX FROM DECIMAL--
260 REM L=LOG(D)/LOG(16):REM LOG(D) BASE16
265 REM FOR M=INT(L)TO0 STEP-1
270 FOR M=3TO0 STEP-1
280 HX$=HX$+MID$(H$,FNMD(D/16 ↑ M)+1,1)
290 NEXTM:Y=16:RETURN
```

This software, available from Omicron Corporation, allows the CBM and PET Computers to use a wide range of RS-232C peripherals.

K&I Netkit II

by Ira Neal

The 'K&I NETKIT II' adds a true RS-232C serial port to CBM and PET computers, allowing the use of a wide range of computers, modems and printers. The 'NETKIT II' is contained on a small printed circuit board that mounts inside the computer.

The 'NETKIT II' is initialized by a 'sys' command. After the 'sys' command is entered, eleven new commands are wedged into BASIC. These commands enable the user full access to the RS-232C port via this extension to CBM BASIC, known as SERIAL BASIC. The 6502 machine code implementing these new commands is contained within an EPROM on the NETKIT board. The command format is very similar to CBM BASIC. The input/output commands addressed to the IEEE-488 bus are postfixed with a '#' symbol (e.g., PRINT#3,"XYZ..."). SERIAL BASIC command are prefixed with a '!' to differentiate them from standard CBM BASIC (e.g., !PRINT "HELLO"). The added BASIC commands are:

!LIST Lists a CBM BASIC program in memory to the device connected to the RS-232C port of the NETKIT.

!PRINT Causes an expression to be transmitted via the RS-232C port.

!INPUT Looks for characters received via the RS-232C port.

!GET Similar to !INPUT, but returns only a single character at a time.

!OPEN Turns the CBM into a conventional VDU terminal. The bit rate, word length, parity and many other attributes may be altered to operate with a wide variety of other machines.

!ON Transforms the RS-232C port into a second keyboard for the CBM; this allows the CBM to be remotely controlled via the RS-232C port.

!CLOSE Cancels the !ON command.

!LOAD Allows the user to load BASIC programs, commands and data from another computer using a dialect similar to BASIC OR CBM BASIC. They may be done at high speed (up to 4800 baud).

!READ Translates the programs obtained by the !LOAD command into CBM internal format, so they may be executed.

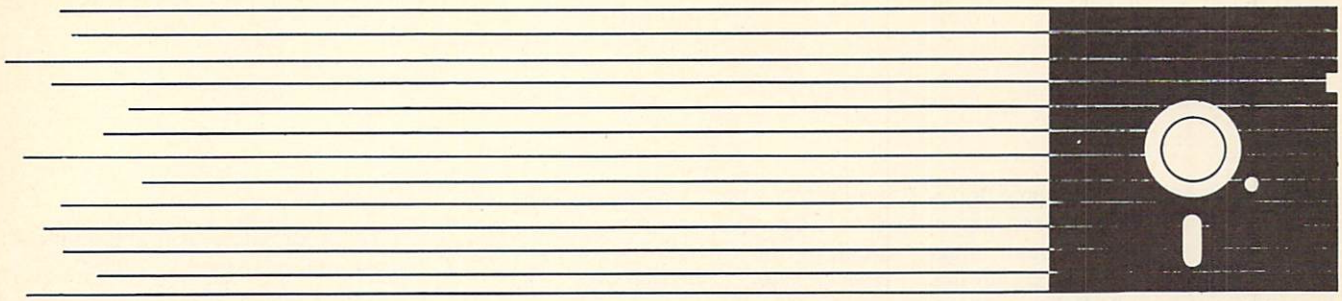
!VERIFY Allows the user to change link bit rate to any value between 50 and 19,200 bits per second.

!KILL Disables SERIAL BASIC and effectively removes all NETKIT features from the system.

The interface contains conversion tables which allow the characters passing through the interface to be replaced by characters more likely to be understood by the system attached to the CBM. This is especially useful in obtaining sensible listings of CBM BASIC programs on a non-CBM printer. The code conversion tables may be altered by the user.

The NETKIT comes with a comprehensive manual that covers installation, operation and testing, and a diskette with three programs. The programs are "NETSUP," a support that aids in setting baud rate and other transmission parameters. "NETTAB," a program to aid in altering the conversion tables, and "BULLETIN.BAS" a computer bulletin board program.

NETKIT II is available from OMICRON CORPORATION, 1416 Providence Highway, Norwood, MA 02062.



DISK QUICKIES

Elizabeth Deal, Malvern, PA

A floppy that comes in the mail or one that has sat around in a box for a long time may seem unreadable. Be patient: put it in the drive and leave it alone. Later, when it is nice, hot and dry, it just may give you all the data it contains. This sounds like rumor, but seems to make sense (Kodak does it for slides!). January and July floppies just cannot be the same thing, unless you let them be the same by not rushing things.

A disk is smart. On 4040 drives and later, when you replace a program with a newer version (SAVE"@1:NEWER",8) it doesn't replace anything at all. The NEWER is stored first in a different place. The old version is scratched only if the write was successful. In the event of a disk full error, or any other disk error, your previous version is intact. You can validate the disk and all will be well. Incidentally, this is one reason why to save with replace you need enough free blocks for the entire NEWER program. "Replace" and "scratch" are misnomers leading to confusion.

Occasionally, for what seems to be no good reason at all, I get a DEVICE NOT PRESENT ERROR, and there seems to be no way out. I found a way out that does not require resetting anything: trying to open a nonexistent file sets things back to normal:

OPEN3,8,3,"xcxcxcxcx" does the trick.

Raeto West of *Programming the PET/CBM* fame sternly recommends against using the write-protect tabs in Commodore disk units. The reason is a bug in the system, whereby the damage done to the floppy might be worse than a scratched file. I won't tell you the details—get the book for it is superb!

If you have BASIC4, you have an alternate set of commands for disk, words such as DOPEN and DSAVE exist. Drive is usually written D0 or D1. You need to be consistent in your use of the commands, SAVE"D1:PROGRAM" will not work, SAVE"1:PROGRAM" and DSAVE"PROGRAM",D1 will. BASIC4 commands are ultimately reduced to the older version's language by the interpreter. Most are described on the last page of the disk book. You might as well use the older commands to be compatible, and to avoid strange secondary address defaults. With the exception of catalogues, error channel and relative file works, BASIC4 disk commands add little to your PET except a bit of confusion.

Happy news for Upgrade PETs with POWER (Professional Software) and POWAID2 (Brad Templeton's additional commands—public domain): you are in a better position than BASIC4 people in terms of disk commands in that the directory and error channel are, as in 4.0, available within program mode, but the directory acts as the corresponding Universal Wedge command in that it can perform a search. You can do this by such statements as SYS(x)"$1", SYS(x)"$1:file*", SYS(x)"$1:*=seq" for the catalogues, SYS(y) for the error channel. X and y can be found in a jump table relatively early in the program (add one to the address). Unlike the wedge program, POWAID is written as a set of self contained subroutines, all have RTS at the end in some form.

This is a powerful way to build up your system (both BASIC and machine language) without the usual, severe time penalty involved in adding BASIC commands. Wouldn't it be wonderful if the future non-4.0-Commodore computers (VIC, 64) contained subroutines such as these?

Coding disk works into machine language is simple once you get the hang of it and when you follow Jim Butterfield's suggestion of using the standard ROM routines. Open the files and channel 15 in BASIC (or machine code or use POWAID or BASIC aid routines) and then simply switch between them: JSR $FFCC, LDX filenumber, JSR$FFC6 or $FFC9, JSR $FFCF or $FFD2 type of cycle handles data files and channel 15 quite effortlessly. The addresses of these ROM routines are in the jump table at the end of memory.

It is sometimes desireable to place most frequently used programs in the beginning of a floppy. One way to reserve some space is to write a short (less than 254 bytes) program, such as "10 PRINT" and save it several times under different names. For instance "a-----","b-----" etc. To place a real file up front, scratching the "a-----" filler makes room for it. I know of no other way of organizing a disk without rewriting one.

To delete a record from a relative file writing CHR$(255) in the first position does the job, you can then recycle the record.

Kids can use their own or your floppies. To keep the house clean I use this procedure: they have two floppies, one ▶

in each drive, basically the same information goes to both drives, hence automatic backup without duplicating and conflicting IDs. I help set up the file names using POWER's REM macros: 10 REM"S=SAVE"@1:HORRORS",8 with 12 left cursors which position the cursor on the "1". All that is then required is for a child to overtype the drive number for the second save.

No, the disk name need not be unique. The only thing that should be unique is the two-character ID. Of course it is better for your own management to give different names. In case of running a childproof "mate" system you may wish to name your floppies "john10/82(j2),j1" and "john (j1),j2" where seeing one tells you which disk is its mate. It seems to work for kids around here.

Three Disk Puzzles

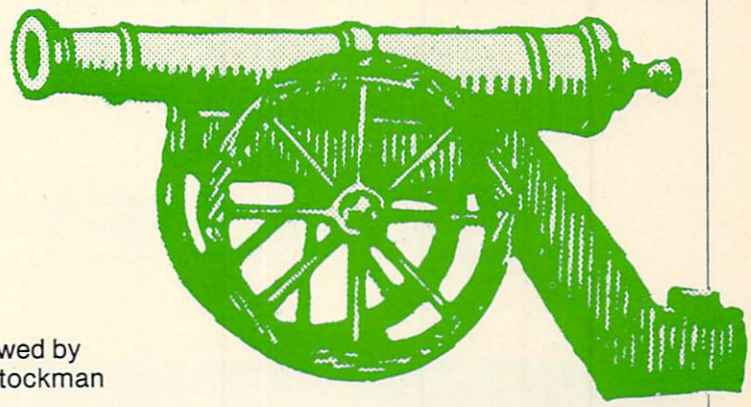
1. Could someone please explain the often-discussed problems in SAVE with replace? I've been using the command, seemingly without trouble and would love to hear where the alleged problems lurk.

2. Why does MLM LOAD to unusual places load only one byte? This weird thing only happens with disk; tapes always load the entire file.

3. Why, when we ask PET to LOAD"1:PROGRAM",8 and PROGRAM is not on the floppy we must push the stop-key to be able to interrogate the error channel while all other disk errors are available without the use of a stop-key?

The Commander

A Utility Package for the PET and CBM



reviewed by
John Stockman

The Commander is not just another toolkit product, it is a great development tool for a BASIC application or a business system. It consists of one chip, a 4K EPROM, containing machine-language programs for linking, chaining, and overlaying multiple BASIC programs as well as high-speed loading and saving of data.

There are two versions of the Commander, so that the chip can be placed in either of 2 sockets within your computer: version 12-3 goes in socket 12 (location $9000 in memory), and version 11-4 goes in socket 11 (location $A000 in memory). The two versions are identical except for the 4K offset in memory addresses.

The Commander system command summary is listed below:

Command	Purpose
INSERT	Insert specific line numbers into a currently running program.
APPEND	Append a program to the end of a currently running program.
DELETE	Deletes a specific line range from a currently running program.
COMMON	Declares specific variables to be preserved when switching programs.
PRINT USING	Formats numeric output.
CONVERT	Converts characters to true ASCII.
FRAME	Allows the program to print a framed message while disk I/O is performed.
OVERLAY	Lets a program of any length overlay the current program.
ENHANCED GET	An advanced keyboard input routine with menu options.
COMPUTED GOTO	Allows the line number to be a computed value.
WINDOW	Easily allows the user to define windows for processing on the screen.
MAT PRINT#	High speed output of data to disk.
MAT INPUT#	High speed input of data from disk.
MAT INIT	Nulls a matrix.
MAT ZER	Sets all values in matrix to 0.
STRING	Inputs a string with punctuation from disk drive.

The Commander is quite easy to use. It also requires no RAM memory and uses only "safe" variable locations while performing its functions. The only set-up required for using the Commander is the creation of specific variables required for the routines, then a SYS to the routines.

EXAMPLE: The INSERT routine.

VARIABLES USED: BL = beginning line
EL = ending line
GT = GOTO line when done
SL = start line of insert program

First fill the variables needed, then SYS 40000 (for Commander 12-3) and a program will be inserted into the current program.

I used the Commander on my CBM 8032 for several days and found it to be quite useful for developing modular BASIC business systems. The manual seems light on real information, but fortunately the demonstration disk which came with the package I evaluated was quite thorough and gave me some good ideas on utilizing the Commander functions.

For availability information, see this issue's New Products section.

SuperPET Update

by
Walt Kutz

This article will diverge from its original charter out of necessity. There have been many requests for information concerning the memory layout and RS-232C wiring diagram of the SuperPET. There are some questions that can best be answered using information directly from the source, namely the folks at Waterloo. The following is taken directly from *SuperPET Specifics*.

Memory Layout in the SuperPET.
The SuperPET has 32K non-bank-switched user memory (random access memory—RAM). The Waterloo microSystems software uses certain areas of this memory space.

In general, the address range from $0000 to $05FF is used entirely by the Waterloo microSystems languages, such as Waterloo microBASIC. (Note: the $-sign in front indicates hexadecimal base). Also, the languages run a stack from $09FF down to $0600. The rest of memory, $0400 to $7FFF, is utilized by each of the languages in some manner.

When these languages are not being used, the following areas may be used as required. The default stacks used by the languages may now be used for user programs. The segment $0A00 to $7FFF is free for program and/or data. As well, some areas of zero page (defined as $0000 to $00FF) may be taken over by the application program to provide less expensive addressing modes both in terms of memory space and execution speed. The low memory segments $0040 to $0060 and $0098 and $00FF are unused in this version of Waterloo microSystems. In addition, memory locations $0080 to $0097 may be used if the Waterloo microSystems floating point routines are not invoked by a user application program.

Reserved Memory of the SuperPET.
This section states only a few of the low memory areas that may be used to your advantage. It is not meant to be complete in any way. Proper usage of these memory areas may help your programs be more portable across future releases.

The two word locations, $20 - $21 and $22 - $23, contain the memory lists available for application programs. The first, referred to as MemBeg-, contains the beginning of available memory address, $0A00, while the second, referred to as MemEnd-, contains the end of memory address, $7FFF.

Any program developed using the Waterloo 6809 Assembler system may be invoked from the main selection menu. By simply entering the name of the program, it will be loaded and execution will begin.

A program may simply issue a return (rts) to go back to the main menu level. Before this is done, the byte location $32, referred to as Service-, must be set to zero, meaning quit. If this is not done, the program will be re-executed.

Handling Interrupts
User programs may want to handle interrupts, such as IRQ. This may be done using the Waterloo microSystems Library routine ConInt- as described in the Waterloo 6809 Assembler literature. To do this properly, your interrupt routine should service only interrupts it recognizes. If the interrupt is unrecognized, control should be passed to the former interrupt processing routine. The soft copy of the interrupt vector is kept at location $0100 to $010D. To find the address of the current interrupt handler, simply index location $0100, known as IntVctr-, by the proper offset.

SuperPET RS-232C Wiring Diagram
The SuperPET uses the following pin connections of the RS-232C port.

Pin	Definition	Symbol
1	Protective ground	
2	Transmitted Data	TxD
3	Received Data	RxD
4	Request to Send	RTS
5	Clear to Send	CTS
6	Data Set Ready	DSR
7	Signal Ground	

8	Data Carrier Detect	DCD
20	Data Terminal Ready	DTR

Some hosts require only connections to TxD, RxD and the ground pins. The SuperPET, however, requires signals at all nine pin connections. To satisfy this requirement, when only pins 1, 2, 3, and 7 are connected to the host, the pins in the SuperPET must be connected according to the following considerations.

In the RS-232C port, the CTS must be at the right voltage level for the transmitter (SuperPET) to work. If the CTS line is not provided from the connecting host cable, the correct level is achieved by connecting the RTS directly to CTS: thereby, the signal going out to pin 4 goes directly back into pin 5 indicating that it is "clear to send"

Similarly, the SuperPET expects to receive the signals DSR and DCD, i.e., that the connecting cable exists. Again, if there are no pin connections from the host cable, the correct voltage is achieved by connecting DRS and DCD directly to DTR. When the SuperPET indicates DTR on pin 20, this signal feeds pins 6 and 8 of the SuperPET, thus returning DRS and DCD respectively.

This "short circuiting" of these pin connections allows the correct voltage levels to be maintained within the microcomputer.

* * *

I thought that I would include, plagiarized a little, an article on how some teachers are using the SuperPET. Portions of this article appeared in the *PERSPECTIVE* a local publication for the residents of the Scotch Plains-Fanwood, NJ, school district. It is interesting to note that these SuperPETs are being used in the middle school and high school, not the colleges and universities as one would expect.

The Teacher's PET is a SuperPET for two teachers, Charles Waters, science teacher at the Scotch-Plains Fanwood High School and Tama Traberman, eighth grade social studies teacher and middle school social studies department chairperson at Terrill Middle School. One teacher is using APL to further learning in biology and the other is focusing on global studies. These innovative uses of the microcomputer have been recognized by Commodore Business Machines, Inc. with their designation of both teachers' classes as Education Resource Centers.

Although other programming languages are available on the SuperPET, both instructors are using APL. Mrs. Traberman and Mr. Waters are fond of APL because it is concise and powerful. APL is also capable of generating rich and varied visual images. These visuals, both teachers believe, make the learning of complex concepts and generalizations easier for students. What makes APL-generated visuals different from other graphics packages is that the students generate their own visuals using data they have researched and collected themselves. These kind of activities, according to Mrs. Traberman and Mr. Waters, involve students in their own learning in a most meaningful way. Students are collecting and analyzing their own data and observations instead of passively reading and interpreting data other people have compiled.

In Mrs. Traberman's classes, students begin by learning the continents, their relative sizes and their total populations. Students enter data about the sizes of the continents into the computer and can then examine the relative sizes by arranging the continents in size order. Populations for each of the continents is researched and entered. Students can then begin to make generalizations about population distribution and population density. Ultimately, Mrs. Traberman plans to explore with students such complex concepts as Per Capita Income, Gross National Product and literacy, all of which are significant in leading students to analyses of relative wealth. Students will learn these concepts in connection with their study of the people and culture of the non-western world.

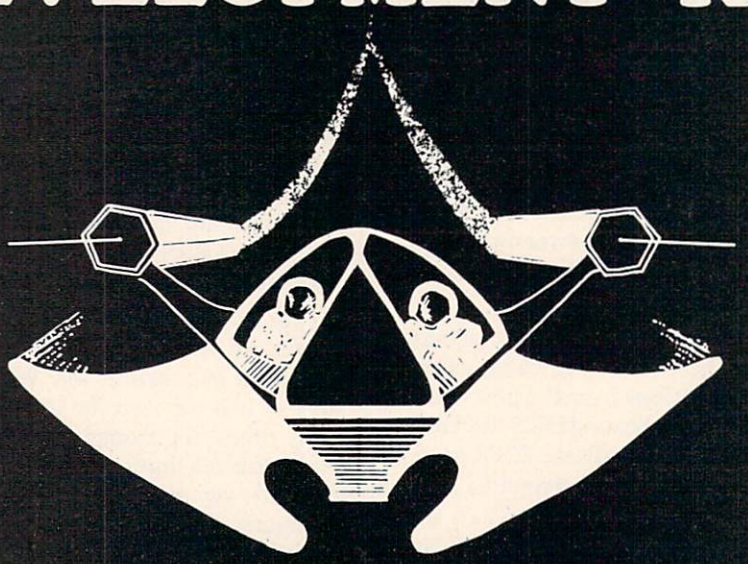
As far as Mr. Waters is concerned, the microcomputer is a marvelous aid for instruction. As students begin to explore problems in genetics, Mr. Waters has them simulating the toss of two coins on the computer. If two coins are tossed forty times, for example, we would expect that we will get two heads ten times, one head and one tail twenty times, and two tails ten times. Instead of actually tossing the coins, the computer simulates the work. In essence, it will toss coins forever if you ask it to do this, will never drop a coin or have to wait for it to stop rolling, and will never make a mistake, and never get bored. When students use the results that the coin toss simulation produces and apply it to the study of genetics, it is an easy matter to see that two heads and two tails can be thought of as homozygous genes, while one head and one tail can be thought of as heterozygous genes. Students in class also observe and record results from the actual mating of fruit files and compare their findings to what might normally be expected according to the laws of genetics. When the results students get are entered into the computer, a CHI square formula determines whether or not their findings differ significantly from what might be expected. Should such significant differences actually turn up, students are led to re-examine their original hypotheses, their actual experimental observations, and sometimes to do the experiment over. The ease of handling large populations by the computer makes such re-examination and reflection a joy rather than a chore.

If you would like further information on exactly how these instructors are implementing the SuperPETs to teach their respective subjects, send a self addressed, postage paid envelope to me. I will return to you reprints of their articles.

If you have any questions concerning the SuperPET or something that you think would be of interest to other SuperPET users please send in your questions or article. I will make a concerted effort to answer any and all questions, space permitting. Please include your telephone number as some questions can be answered without the need for correspondence.

Walt Kutz
SuperPET Product Manager, U.S.
487 Devon Park Drive
Wayne, PA 19087

GAME PROGRAM DEVELOPMENT KIT



for the

VIC 20™ and COMMODORE 64™

Registered trademarks of Commodore Business Machines, Inc.

SIX TOOLS TO HELP YOU WRITE YOUR OWN FAST ACTION ARCADE-STYLE GAMES

DECODER — Decodes programs written in machine language (like game cartridges, utility cartridges, and even the computer's own internal operating programs). Produces a program in an English-like language (Assembler) which can be studied to figure out how they did it. The programs created with the decoder can be customized with the **EDITOR** AND INCORPORATED INTO YOUR OWN NEW GAME PROGRAM. The **ASSEMBLER** turns your programs created with the Decoder and the Editor back into machine language and puts them out to tape or disk so the **LOADER** can load them into the computer's memory to be tested and RUN. The **MONITOR** assists you in debugging your new game program by allowing you to run it a step at a time and making modifications if you need to. The **INSTRUCTION GUIDE** is written so that even a beginner can learn the skills needed to become a pro!!!

DESIGNED TO RUN ON ALL VIC-20's

$49.95 plus $2.00 p&h buys the kit that could make you rich. Why wait?

Send check, M.O., VISA/MC ($2.00 s.c., please include expiration date), or specify COD (add $3.00) to:

French Silk smooth ware

P.O. Box 207, Cannon Falls, MN 55009

507-263-4821

NEW PRODUCT DEVELOPMENTS

Company:
Commander Systems, Inc.
4505 Jackson Street
Hollywood, FL 33021
305-962-5183

Product:
The Commander—4K ROM for use with Commodore 2000, 4000 and 8000 series. Provides exclusive programmable commands formerly available only on large systems. Under program control or direct mode, these commands contain an enhanced COMMON function that retains all variables and arrays. Commands allow the user to insert a disk program or subroutine between line numbers without losing variables or arrays, append another program or subroutine to the end of the running program, delete any portion of a running program, format a floating point variable into a string, delete all GOSUBs and FOR... NEXT loops from the operating system, write an entire array to the disk or input an entire array from the disk and several other operations with ease.

Price: $70.00

Company:
United Microware Industries, Inc.
3503-C Temple Avenue
Pomona, CA 91768
714-594-1351

Product:
Wordcraft 20©—Wordprocessor for the VIC 20. A fully featured cartridge program with 8K of RAM, Wordcraft 20 displays documents on the screen and prints them exactly as typed, so editing is visually simplified. Automatic four-direction scrolling permits complete viewing of the 99-character by 66-line page capacity. Characters, words and large copy blocks can be deleted, inserted, moved and copied and up to 40 pages can be stored on disk or tape. A writer can also personalize business letters using Wordcraft 20's mailing list files. If electronic mail is sent, it can be made unreadable to anyone without a keyword.

Price: $269.95

Company:
Info-Designs
6905 Telegraph Road
Birmingham, MI 48010
313-540-4010

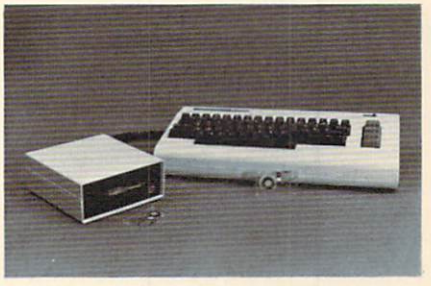
Product:
Management Accounting System for Commodore 64—Scaled-down versions of the Info-Design Management Accounting System for CBM computers. The first three modules to be released include accounts receivable/billing, accounts payable/check writing, and general ledger. Can be used with one or two 1541 single disk drives and a matrix printer.

Price: $199 for each module

Company:
Exatron
181 Commercial Street
Sunnyvale, CA 94086
408-737-7111

Product:
ESF-20/64 Stringy Floppy Mass Storage System—For the VIC 20 and Commodore 64. Consists of a miniature endless-loop tape cartridge and an electronic direct-drive transport mechanism that plugs into the computer's serial bus connector. The tape transport operates at a speed of 5 inches per second and has a memory capacity up to 64K bytes. Previously stored data is easily transferrable to the ESF-20/64 system.

Price: $199.50



Company:
Custom Software Design
831 Maplewood Avenue
Anderson, IN 46012

Product:
STOCK—Investment analysis for PET 4032, CBM 8032 and Commodore 64. STOCK provides comprehensive statistical analysis of a single stock, storage and retrieval of up to 150 stocks per disk, side-by-side screen comparison of any two stocks, screen graphical display of selected factors, printing of single stock analyses and printing of comparison of two to five stocks. Ideal for use by investment clubs, personal finance or consumer education classes and personal investors.

Price: $95.00

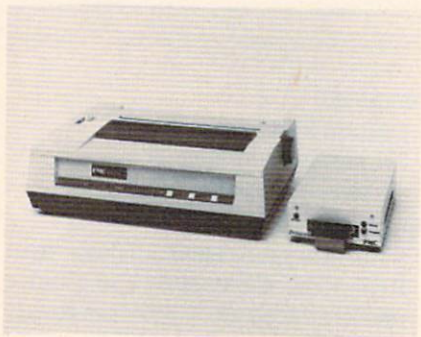
NEW PRODUCT DEVELOPMENTS

Company:
RAK Electronics
P.O. Box 1585
Orange Park, FL 32067

Product:
CW MORSE—Morse code terminal for PET 2000/4000 series with 8K or more memory, unexpanded or expanded VIC 20, and Commodore 64. Software is written in BASIC for ease of modification. Requires construction of a two-transistor/one-IC interface. Connection is made through the I/O user port. Package includes software on cassette, complete documentation, interface schematic and connector.

Price: $19.95

Company:
Personal Micro Computers, Inc.
475 Ellis Street
Mountain View, CA 94043
415-962-0220



Product:
PrinterMaid™—64K universal intelligent buffer. Allows either parallel or serial interface between computer and printer. The PrinterMaid accepts data over the parallel interface at up to 3,600 bytes per second and transfers data on the serial interfaces at 50 to 19,200 baud. If the message to be printed is less than 64,000 characters long, the PrinterMaid accepts the entire message, releases the computer and takes over the task of supplying characters to the printer. It is both hardware and software programmable and can use any of four serial protocols: CTS/RTS, DSR/DTR, XON/XOFF and ETX/ACK.

Price: $399.00

Company:
Alfred Publishing Co., Inc.
15335 Morrison Street
P.O. Box 5964
Sherman Oaks, CA 91413
213-995-8811

Product:
Handy Guides—New titles include: *How to Buy a Word Processor* by Michael Scriven, *How to Use VisiCalc/SuperCalc* by Carlton Shrum, *Understanding Data Base Management* by Michael J. Freiling, *Understanding APL* by Susan M. Bryson, and *Understanding LISP* by Paul Y. Gloess.

Price: $2.95 each

Company:
Human Engineered Software
71 Park Lake
Brisbane, CA 94005
415-468-4110

Product:
Four new languages/utilities for VIC 20—1) VIC FORTH: an interactive, high-level language many times faster than BASIC. 2) HES Writer: a word processor with full screen editing, word wrap-around, justification, text centering, page numbering and more. 3) HES MON: a 6502 machine language monitor with a mini assembler. Has nearly 30 commands. Includes forward and reverse scrolling of disassembly, hex and ASCII memory dump displays, soft recovery from programming errors in full-speed execution mode and more. 4) Turtle Graphics: an easy-to-use language that introduces users to computers and computer concepts. Mathematical, logical, communications and artistic skills are taught by moving an imaginary turtle on the screen.

Price: VIC FORTH—$59.95; HES Writer—$39.95; HES MON—$39.95; Turtle Graphics—$39.95.

USER CLUBS
Sound Off!

We're continuing to compile a list of all Commodore Users clubs throughout the country. If you'd like to add your name to the rolls, please send your club's name, address, and other pertinent information to:

Commodore Users Clubs
c/o Editor
Commodore Magazine
487 Devon Park Drive
Wayne, PA 19087

ALABAMA

Huntsville PET Users Club
9002 Berclair Road
Huntsville, AL 35802
Contact: Hal Carey
Meetings: every 2nd Thursday

ARIZONA

VIC Users Group
1206 N. Fraser Drive
Mesa, AZ 85203
Contact: Paul V. Muffuletto

Commodore User Group
Metro Computer Store
4500 E. Speedway,
Suite 13
Tucson, AZ 85712
602-323-3116

Central Arizona PET People
842 W. Calle del Norte
Chandler, AZ 85224
602-899-3622
Roy Schahrer

ACUG
c/o Home Computer Service
2028 W. Camelback Rd.
Phoenix, AZ 85015
602-249-1186
Dan Deacon
First Wed. of month

ARKANSAS

Commodore/PET Users Club
Conway Middle School
Davis Street
Conway, AR 72032
Contact: Geneva Bowlin

Booneville 64 Club
c/o A.R. Hederich
Elementary School
401 W. 5th St
Booneville, AK 72927
Mary Taff

CALIFORNIA

Lawrence Hall of Science
UC Berkeley
Computer Project, Room 254
Berkeley, CA 94720
(415) 527-9286

PALS (PETS Around Livermore Society)
886 South K
Livermore, CA 94550
Contact: John Rambo

SCPUG Southern California PET Users Group
c/o Data Equipment Supply Corp.
8315 Firestone Blvd.
Downey, CA 90241
(213) 923-9361
Meetings: First Tuesday of each month

California VIC Users Group
c/o Data Equipment Supply Corp.
8315 Firestone Blvd.
Downey, CA 90241
(213) 923-9361
Meetings: Second Tues. of each month

Commodore Users Club
1041 Foxenwoods Drive
Santa Maria, CA 93455
(805) 937-4106
Contact: Greg Johnson

Valley Computer Club
2006 Magnolia Blvd.
Burbank, CA
(213) 849-4094
1st Wed. 6 p.m.

Valley Computer Club
1913 Booth Road
Ceres, CA 95307

PUG of Silicon Valley
22355 Rancho Ventura Road
Cupertino, CA 95014

BAMBUG
1450 53rd Street
Emeryville, CA
(415) 523-7396

North Orange County Computer Club
3030 Topaz, Apt. A
Fullerton, CA 92361
Dave Smith

Lincoln Computer Club
750 E. Yosemite
Manteca, CA 95336
John Fung, Advisor

PET on the Air
525 Crestlake Drive
San Francisco, CA 94132
Max J. Babin, secretary

PALS (PETs Around Livermore Society)
886 South K
Livermore, CA 94550
(415) 449-1084
Every third Wednesday
7:30 p.m.
Contact: J. Johnson

SPHINX
7615 Leviston Ave.
El Cerrito, CA 94530
Bill MacCracken

San Diego PUG
c/o D. Costarakis
3562 Union Street
(714) 235-7626
7 a.m.-4 p.m.

Walnut Creek PET Users Club
1815 Ygnacio Valley Road
Walnut Creek, CA 94596

Jurupa Wizards
4526 Kingsbury Pl.
Riverside, CA 92503
Contact: Walter J. Scott

Vincent Yanniello's VIC 20 Software Exchange Club
2130 Colby Avenue
West Los Angeles, CA 90025
(213) 479-3000

The Commodore Connection
2301 Mission St.
Santa Cruz, CA 95060
408-425-8054
Bud Massey

64 Users Group
6029 Conta Costa Rd.
Oakland, CA 94618
E. R. Kennedy

San Fernando Valley Commodore Users Group
21208 Nashville
Chatsworth, CA 91311
213-709-4736
Tom Lynch
2nd Wed. 7:30

COLORADO

VICKIMPET Users' Group
4 Waring Lane, Greenwood Village
Littleton, CO 80121
Contact: Louis Roehrs

CONNECTICUT

John F. Garbarino
Skiff Lane Masons Island
Mystic, CT 06355
(203) 536-9789

Commodore User Club
Wethersfield High School
411 Wolcott Hill Road
Wethersfield, CT 06109
Contact: Daniel G. Spaneas

VIC Users Club
c/o Edward Barszczewski
22 Tunxis Road
West Hartford, CT 06107

New London County Commodore Club
Doolittle Road
Preston, CT 06360
Contact: Dr. Walter Doolittle

FLORIDA

Jacksonville Area PET Society
401 Monument Road, #177
Jacksonville, FL 32211

Richard Prestien
6278 SW 14th Street
Miami, FL 33144

South Florida PET Users Group
Dave Young
7170 S.W. 11th
West Hollywood, FL 33023
(305) 987-6982

VIC Users Club
c/o Ray Thigpen
4071 Edgewater Drive
Orlando, FL 32804

PETs and Friends
129 NE 44 St.
Miami, FL 33137
Richard Plumer

Sun Coast VICs
P.O. Box 1042
Indian Rocks Beach, FL 33535
Mark Weddell

Bay Commodore Users Group
c/o Gulf Coast Computer Exchange
241 N. Tyndall Pkwy
P.O. Box 6215
Panama City, FL 32401
904-785-6441
Richard Scofield

IDAHO

GHS Computer Club
c/o Grangeville High School
910 S. D St.
Grangeville, ID 83530
Don Kissinger

S.R.H.S. Computer Club
c/o Salmon River H.S.
Riggins, ID 83549
Barney Foster

ILLINOIS

Shelly Wernikoff
2731 N. Milwaukee Avenue
Chicago, IL 60647

VIC 20/64 Users Support Group
c/o David R. Tarvin
114 S. Clark Street
Pana, IL 62557
(217) 562-4568

Central Illinois PET User Group
635 Maple
Mt. Zion, IL 62549
(217) 864-5320
Contact: Jim Oldfield

ASM/TED User Group
200 S. Century
Rantoul, IL 61866
(217) 893-4577
Contact: Brant Anderson

PET VIC Club (PVC)
40 S. Lincoln
Mundelein, IL 60060
Contact: Paul Schmidt, president

Rockford Area PET User's Group
1608 Benton Street
Rockford, IL 61107

Commodore Users Club
1707 East Main St.
Olney, IL 62450
Contact: David E. Lawless

VIC Chicago Club
3822 N. Bell Ave.
Chicago, IL 60618
John L. Rosengarten

DuPage-64
708 Prairie Ave.
Glen Ellyn, IL 60137
312-858-0637
Gus Pagnotta

Oak Lawn Commodore Users Club
The Computer Store
11004 S. Cicero Ave.
Oak Lawn, IL 60453
Bob Hughes

INDIANA

PET/64 Users
10136 E. 96th St.
Indianapolis, IN 46256
317-842-6353
Jerry Brinson

Cardinal Sales
6225 Coffman Road
Indianapolis, IN 46268
(317) 298-9650
Contact: Carol Wheeler

CHUG (Commodore Hardware Users Group)
12104 Meadow Lane
Oaklandon, IN 46236
Contact: Ted Powell

VIC Indy Club
P.O. Box 11543
Indianapolis, IN 46201
(317) 898-8023
Ken Ralston

IOWA

Commodore User Group
114 8th St.
Ames, IA 50010

Quad City Commodore Club
1721 Grant St.
Bettendorf, IA 52722
319-355-2641
John Yigas

Commodore Users Group
965 2nd St.
Marion, IA 52302
319-377-5506
Vern Rotert
3rd Sun. of month

KANSAS

Wichita Area PET Users Group
2231 Bullinger
Wichita, Kansas 67204
(316) 838-0518
Contact: Mel Zandler

Kansas Commodore Computer Club
101 S. Burch
Olathe, KS 66061
Contact: Paul B. Howard

Commodore Users Group
6050 S. 183 St. West
Viola, KS 67149
Walter Lounsbery

VIC Connection
1010 S. Elm
Henderson, KY 42420
Jim Kemp

KENTUCKY

VIC Connection
1010 S. Elm
Henderson, KY 42420
Jim Kemp

LOUISIANA

Franklin Parish Computer Club
#3 Fair Ave.
Winnsboro, LA 71295
James D. Mays, Sr.

NOVA
917 Gordon St.
New Orleans, LA 70117
(504) 948-7643
Kenneth McGruder, Sr.

VIC 20 Users Group
5064 Bowdon St.
Marrero, LA 70072
504-341-5305
Wayne D. Lowery, R.N.

MARYLAND

Assoc. of Personal Computer Users
5014 Rodman Road
Bethesda, MD 20016

Blue TUSK
700 East Joppa Road
Baltimore, MD 21204
Contact: Jim Hauff

House of Commodore
8835 Satyr Hill Road
Baltimore, MD 21234
Contact: Ernest J. Fischer

Long Lines Computer Club
323 N. Charles St., Rm. 201
Baltimore, MD 21201
Gene Moff

MASSACHUSETTS

Eastern Massachusetts VIC Users Group
c/o Frank Ordway
7 Flagg Road
Marlboro, MA 02173

VIC Users Group
c/o Ilene Hoffman-Sholar
193 Garden St.
Needham, MA 02192

Commodore Users Club
Stoughton High School
Stoughton, MA 02072
Contact: Mike Lennon

Berkshire PET Lovers
CBM Users Group
Taconic High
Pittsfield, MA 01201

The Boston Computer Society
Three Center Plaza
Boston, MA 02108
(617) 367-8080
Mary E. McCann

VIC Interface Club
c/o Procter & Gamble Inst. Shop
780 Washington St.
Quincy, MA 02169
C. Gary Hall

MICHIGAN

David Liem
14361 Warwick Street
Detroit, MI 48223

VIC Users Club
University of Michigan
School of Public Health
Ann Arbor, MI 48109
Contact: John Gannon

Commodore User Club
32303 Columbus Drive
Warren, MI 48093
Contact: Robert Steinbrecher

Commodore Users Group
c/o Family Computer
3947 W. 12 Mile Rd.
Berkley, MI 48072

VIC Commodore Club
2765 Bristol Rd.
Dowling, MI 49050
Bill Kelley

W. Michigan VIC 20-64 Users
1311 Portland NE
Grand Rapids, MI 49505
616-459-7578
Jim D'Haem

MINNESOTA

MUPET (Minnesota Users of PET)
P.O. Box 179
Annandale, MN 55302
c/o Jon T. Minerich

Twin Cities Commodore Computer Club
6623 Ives Lane
Maple Grove, MN 55369
(612) 424-2425
Contact: Rollie Schmidt

MISSOURI

Clearwater Club
Clearwater School
Star Route
Piedmont, MO 63957
Contact: Carolyn Polk

KCPUG
5214 Blue Ridge Boulevard
Kansas City, MO 64133
Contact: Rick West
(816) 356-2382

PET SET Club of St. Louis
1501 Carman Road
Manchester, MO 63011
(314) 527-3236
Contact: Ed or Til Keil

MONTANA

Powder River Computer Club
Powder River County High School
Broadus, MT 59317
Contact: Jim Sampson

Commodore User Club
1109 West Broadway
Butte, MT 59701
Contact: Mike McCarthy

NEVADA

Las Vegas PET Users
4884 Iron Avenue
Las Vegas, NV 89110

NEW JERSEY

Amateur Computer Group
18 Alpine Drive
Wayne, NJ 07470

Somerset Users Club
49 Marcy Street
Somerset, NJ 08873
Contact: Robert Holzer

Educators Advisory
P.O. Box 186
Medford, NJ 08055
(609) 953-1200
John Handfield

VIC-TIMES
46 Wayne St.
Edison, NJ 08817
Thomas R. Molnar

NEW HAMPSHIRE

Northern New England Computer Society
PO Box 69
Berlin, NH 03570

TBH VIC-NICs
P.O. Box 981
Salem, NH 03079

NEW MEXICO

Commodore Users Group
6212 Karlson, NE
Albuquerque, NM 87113
505-821-5812
Danny Byrne

NEW YORK

Capital District PET Users
Ben Green
Albany Area, NY
(518) 370-1820

Long Island PET Society
Ralph Bressler
Harborfields HS
Taylor Avenue
Greenlawn, NY 11740

PET User Club of Westchester
Box 1280
White Plains, NY 10602
Contact: Ben Meyer

LIVE (Long Island VIC Enthusiasts)
17 Picadilly Road
Great Neck, NY 11023
Contact: Arnold Friedman

Commodore Masters
25 Croton Ave.
Staten Island, NY 10301
Contact: Stephen Farkouh

VIC Users Club
76 Radford St.
Staten Island, NY 10314
Contact: Michael Frantz

VIC Users Club
c/o Christopher Kwasnicki
44 Harvey Ave.
Staten Island, NY 10314

PUG of Rockland County
c/o Ross Garber
14 Hillside Court
Suffern, NY 10901
(914) 354-7439

West Chester County VIC Users Group
P.O. Box 146
Pelham, NY 10552
Joe Brown

SPUG
4782 Boston Post Rd.
Pelham, NY 10803
Paul Skipski

VIC 20 User Club
151-28 22nd Ave.
Whitestone, NY 11357
Jean F. Coppola

VIC 20 User Club
339 Park Ave.
Babylon, NY 11702
(516) 669-9126
Gary Overman

VIC User Group
1250 Ocean Ave.
Brooklyn, NY 11230
212-859-3030
Dr. Levitt

L&M Computer Club
VIC 20 & 64
4 Clinton St.
Tully, NY 13159
315-696-8904
Dick Mickelson

Commodore Users Group
1 Corwin Pl.
Lake Katrine, NY 12449
J. Richard Wright

8*8 Enthusiasts
Box 28 Rhodes Rd.
Apalachin, NY 13732
Keith Merrill

VIC 20/Commodore 64 Users Group
31 Maple Dr.
Lindenhurst, NY 11757
516-957-1512
Pete Lobol

NORTH CAROLINA

Amateur Radio PET User's Group
PO Box 30694
Raleigh, NC 27622
Contact: Hank Roth

Commodore Users Club
4241 Castleton Road
Charlotte, NC 28211
Contact: Ed Harris

VIC Users Club
c/o David C. Fonenberry
Route 3, Box 351
Lincolnton, NC 28092

Microcomputer Users Club
Box 17142 Bethabara Sta.
Winston-Salem, NC 27116
Joel D. Brown

VIC Users Club
Rt. 11, Box 686
Hickory, NC 28601
Tim Gromlovits

OHIO

Dayton Area PET User Group
933 Livingston Drive
Xenia, OH 45385
B. Worby, president
(513) 848-2065
J. Watson, secretary
(513) 372-2052

Central Ohio PET Users Group
107 S. Westmoor Avenue
Columbus, Ohio 43204
(614) 274-6451
Contact: Philip H. Lynch

Toledo PETS
734 Donna Drive
Temperance, MI 48182
Contact: Gerald Carter, president

OKLAHOMA

Southwest Oklahoma Computer Club
4311 Floyd Ave. NW
Lawton, OK 73505
Joe Teeples

OREGON

NW PET Users Group
John F. Jones
2134 N.E. 45th Avenue
Portland, OR 97213

PENNSYLVANIA

PET User Group
Gene Beals
PO Box 371
Montgomeryville, PA 18936

Penn Conference Computer Club
c/o Penn Conference of SDA
720 Museum Road
Reading, PA 19611
Contact: Dan R. Knepp

PACS PET Users Group
20th & Olney Streets
Philadelphia, PA

Glen Schwartz
807 Avon
Philadelphia, PA 19116

Gene Planchak
4820 Anne Lane
Sharpsville, PA 15150
(412) 962-9682

PPG (Pittsburgh PET Group)
c/o Joel A. Casar, DMD
2015 Garrick Drive
Pittsburgh, PA 15235
(412) 371-2882

Bucks County Commodore User Club
72 E. Rambler Dr.
Holland, PA 18966
(215) 322-0394
Jim Dubrouillet

ABEPET Users Group
P.O. Box 6051
Lehigh Valley, PA 18001
Bill Thomas

Westmoreland Commodore Users Club
c/o DJ & Son Electronics
Colonial Plaza
Latrobe, PA 15650
Jim Mathers

PUERTO RICO

CUG of Puerto Rico
RFD #1, Box 13
San Juan, PR 00914
Ken Burch

VIC 20 User Group
655 Hernandez St.
Miramar, PR 00907
Robert Morales, Jr.

RHODE ISLAND

Irving B. Silverman, CPA
160 Taunton Ave.
E. Providence, RI 02914
Contact: Michelle Chavanne

SOUTH DAKOTA

PET User Group
515 South Duff
Mitchell, SD 57301
(605) 996-8277
Contact: Jim Dallas

TENNESSEE

River City Computer Hobbyists
Memphis, TN
1st Mon. at Main Library

Nashville VIC Users Group
American Computer Store
1004 8th Ave., S.
Nashville, TN 37203
(615) 242-8592
Jane Maggard
1st Thurs. of month, 6 pm

Commodore User Club
Metro Computer Center
1800 Dayton Blvd.
Chattanooga, TN 37405
Mondays 7:30 pm

TEXAS

SCOPE
1020 Summit Circle
Carrolton, TX 75006

PET Users
2001 Bryan Tower
Suite 3800
Dallas, TX 75201

Larry Williams
PO Box 652
San Antonio, TX 78293

PET User Group
John Bowen
Texas A & M
Microcomputer Club
Texas A & M, TX

User Bulletin Board

USER GROUPS FORMING

Florida
VIC 20/Commodore 64 User Group forming:
Contact Stephen McHaney
4912 Blarney Drive
Orlando 32808

New York
Commodore 64 User Group forming:
Contact Keith Merrill
Box 28 Rhodes Road
Apalachin 13732

Louisiana
VIC 20 User Group forming:
Contact Wayne Lowery
5064 Bowdon Street
Marrero 70072
504-341-5305

Commodore User Group forming:
Contact Raymond A. Jobes
180 Princess Drive
Rochester 14623
716-334-0588 or 288-8440

User Clubs Continued

CHUG (Commodore Houston Users Group)
8738 Wildforest
Houston, TX 77088
(713) 999-3650
Contact: John Walker

Corpus Christi Commodores
3650 Topeka St.
Corpus Christi, TX 78411
512-852-7665
Bob McKelvy

Commodore Users Group
5326 Cameron Rd.
Austin, TX 78723
512-459-1220
Dr. Jerry D. Frazee

UTAH

Utah PUG
Jack Fleck
2236 Washington Blvd.
Ogden, UT 84401

The Commodore User's Club
742 Taylor Avenue
Ogden, Utah 84404
Contact: Todd Woods Kap, president;
David J. Shreeve, vice president

The VIClic
799 Ponderosa Drive
Sandy, UT 84070
Contact: Steve Graham

VIC 20 Users
324 N. 300 W.
Smithfield, UT 84335
Dave DeCorso

Northern Utah VIC & 64 Users Group
P.O. Box 533
Garland, UT 84312
David Sanders

VIRGINIA

Northern VA PET Users
Bob Karpen
2045 Eakins Court
Reston, VA 22091
(803) 860-9116

VIC Users Group
Rt. 2, Box 180
Lynchburg, VA 24501
Contact: Dick Rossignol

VIC Users Group
c/o Donnie L. Thompson
1502 Harvard Rd.
Richmond, VA 23226

Dale City Commodore User Group
14752 Danville Rd.
Dale City, VA 22193
(703) 680-2270
James Hogler

Tidewater Commodore Users Group
417 Bethune Dr.
Virginia Beach, VA 23452
H. Thomas Baise, III

Fredericksburg Area Computer Enthusiasts
P.O. Box 324
Locust Grove, VA 22508
703-972-7195
Michael Parker

Commonwealth 20/64 Users Group
1773 Wainwright Dr.
Reston, VA 22090
703-471-6325
Tal Carawan, Jr.

WASHINGTON

NW PET Users Group
2565 Dexter N. 3203
Seattle, WA 98109
Contact: Richard Ball

PET Users Group
c/o Kenneth Tong
1800 Taylor Ave. N102
Seattle, WA 98102

VIC 20 Computer Club
947 N. Burroughs Ave.
Oak Harbor, WA 98277
Michael D. Clark

WISCONSIN

Sewpus
c/o Theodore J. Polozynski
PO Box 21851
Milwaukee, WI 53221

Waukesha Area Commodore User Group (WACUG)
256½ W. Broadway
Waukesha, WI 53186
Contact: Walter Sadler
(414) 547-9391

Commodore User Group
1130 Elm Grove St.
Elm Grove, WI 53122
Tony Hunter

Commodore 64 Software Exchange Group
P.O. Box 224
Oregon, WI 53575
E. J. Rosenberg

CANADA

Toronto PET Users Group
381 Lawrence Ave. West
Toronto, Ontario, Canada
M5M 1B9
(416) 782-9252
Contact: Chris Bennett

PET Users Club
c/o Daniel Cayer
R.R. 6
Simcoe, Ontario
Canada N3Y 4K5

Vancouver PET Users Group
Box 91164
West Vancouver, British Columbia
Canada V7V 3N6

CCCC (Canadian Commodore Computer Club)
c/o Strictly Commodore
47 Coachwood Place
Calgary, Alberta, Canada
T3H 1E1
Contact: Roger Olanson

W.P.U.G.
9-300 Enniskillen Ave.
Winnipeg, Manitoba R2V 0H9
Larry Neufeld

VIC-TIMS
2-830 Helena St.
Trail, British Columbia V1R 3X2
604-368-9970
Greg Goss

Arva Hackers
Medway High School
Arva, Ontario N0M 1C0
D. Lerch

KOREA

Commodore Users Club
K.P.O. Box 1437
Seoul, Korea
Contact: S. K. Cha

MEXICO

Asociacion De Usuarios Commodore
c/o Alejandro Lopez Arechiga
Holbein 174-6° Piso
Mexico 18, D.F.

Club de Usarios Commodore Sigma del Norte
Mol del Valle, Local 44
Garza Garcia, N.L. 66220

NEW ZEALAND

Commodore Users Group
Meet at VHF Clubrooms
Hazel Ave.
Mount Roskill
3rd Wed of month, 7:30 pm
Roger Altena 278-5262

Nelson VIC Users Group
c/o P.O. Box 860
Nelson, New Zealand
Peter Archer

NORWAY

VIC Club of Norway
Nedre Bankegt 10,
1750 Halden
Norway

UNITED KINGDOM

North London Hobby Computer Club
Dept. of Electronics & Communications Engineering
The Polytechnic of North London
Holloway Rd.
London N7 8DB

Croydon Microcomputer Club
111 Selhurst R.
Selhurst, London SE25 6LH
01-653 3207
Vernon Gifford

MIS VIC-20™ SOFTWARE

WARP RUNNER©

100% MACHINE 100% FAST

WARP RUNNER is a multi-color hi-res arcade game that runs on the unexpanded VIC-20. JOYSTICK REQUIRED.

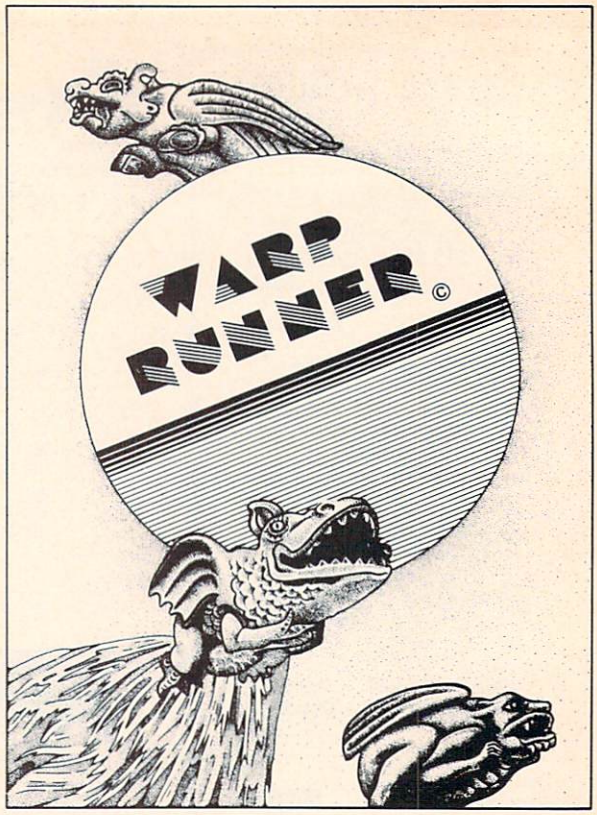
A report by MIS Software Test Pilot Tony Scioppazi, the Bronx, New York.

Look, bud, if you want to be a Warp Runner, you've got to be good, right? You've got to be calm and steady, but you've got to be quick. You know what I mean? You've got to be able to jockey those sub-light cuuisers just like a Bronx cabbie during rush hour, in and out. Do you catch my drift? Don't even THINK about running into anything. Put even a scratch on one of those babies and its a strike on you. Three strikes, and you're yesterday's news—strictly out. Are you with me? Just make sure you don't zig when you should have zagged.

Aim those laser blasts carefully. A missed shot could be bad for business. Get right up on those pigeons; shorter shots are faster shots. Don't worry about blasting everything in sight; just get through. You read me? Good luck!

WARP RUNNER is available on cassette with complete documentation, enclosed in an attractive vinyl binder for $16.95 each.

MIS produces the finest educational, recreational, and functional software available for the Commodore VIC-20 Personal Computer. Ask for MIS software at your local computer store or order direct from MIS. VISA, MASTER-CARD, phone orders, and C.O.D. accepted. California residents add sales tax.



250 FERN ROCK WAY • BOULDER CREEK, CA • 408-338-9546

COMING SOON! "THE FINAL CONFLICT"
A HI-RES, MACHINE LANGUAGE ARCADE GAME FOR THE 64

YOU DON'T HAVE TO BE A PIRATE *TO AFFORD* TOTL *SOFTWARE*

for the* VIC 20* *and* COMMODORE 64

WORD PROCESSING ***$35.00***
Full capability word processing. Word-oriented—not a line editor. Menu-driven. Outstanding features including: footnotes, headings, footing, keyboard input, special printer control, and more. VIC: TOTL.TEXT 2.5 64: TOTL.TEXT 2.6

MAILING LIST and LABELS ***$20.00***
Easy editing, automatically sorted, optional non-printing data line(s), browse and select functions. Menu-driven. VIC or 64: TOTL.LABEL 2.0

KEYWORD CROSS REFERENCE ***$25.00***
Students and authors: keep track of reference notes and bibliographies. Quick reference by keyword. Requires printer. VIC or 64: RESEARCH ASSISTANT 2.0

TIME MANAGEMENT ***$25.00***
Keep track of activities by date (and time). Screen inquiry by date, person, project. 56 different bar chart formats available. VIC or 64: TOTL TIME MANAGER 2.0

BUSINESS ACCOUNTING ***(To be announced)***
Accounts receivable and payable. Inventory and expense tracking. Print invoices, statements, reports. Disk only.

SPECIAL VERSIONS AVAILABLE for QUANTUM DATA INC. 40/80 COLUMN VIDEO BOARD.

All programs work with disk and/or tape; VIC or RS-232 printers. VIC requires minimum 8K expansion.

LOOK for TOTL Software at your Dealer or order direct—shipping included.

TOTL software

Send check or money order and be sure to specify machine. California residents add 6% sales tax.

P.O. Box 4742 • Walnut Creek, CA 94596

MasterCard VISA Call (415) 943-7877

*VIC 20 and COMMODORE 64 are trademarks of Commodore Business Machines.

FOX 20:™

The magazine for *VIC-20 users

*1982

FOX 20 will provide you, each month, with 5 or more ready-to-run programs on cassette-exciting, imaginative, game, educational, and utility programs for most VIC memory configurations. In additon, you will receive Foxtales, an informative Video Newsletter, on Side 2. Novice and pro alike, FOX 20 is the magazine for you. Be a sly little fox for only $43 a year - $5.25 single issue.

FOX 20 is a division of:
Foxfire Systems, Inc.
3811 Newton
Pasadena, Texas 77503

To order: Send check (U.S. Funds), add 5% Sales Tax for Texas residents, Canada and overseas $53, orders pre-paid. Dealer and author inquiries invited.

FOX 20:™

The magazine for *VIC-20 users

P. O. BOX 507
Deer Park, Texas 77536
(713) 473-6723

*VIC-20 is a trademark of Commodore Business Machines

Captain Scuttlebutt's Unbelievable Rumors

We thought we'd been very tricky, sneaking off to the January Consumer Electronics Show (CES) in Las Vegas without telling our pal Scuttlebutt. But no sooner did we have the booth operating than we noticed someone lurking in the shadows, his huge ears waggling in an ecstasy of rumor collecting. He got back to the office before anyone else and submitted this to us as we were going to press.

Oh, what bliss for a person of my calling! Awash in an ocean of rumors at the Consumer Electronics Show! Let me tell you quickly, since I don't have much time, the things I heard and saw at the Commodore booth. But remember, this is strictly unauthorized. You're really not supposed to know. . .but it's so interesting, I've just got to tell you.

I saw two brand new Commodore computers there—a portable 64K color computer and a hand-held computer—that up until the show were so top secret that not even I had heard about them. The portable is the hot one, they say, with a built-in monitor (color or black-and-white, depending on what you need) and a choice of either one or two disk drives, also built-in. According to reliable sources, it folds up into a tidy little briefcase you can take anywhere. Not only that, but it's supposed to be compatible with the Commodore 64, and retail for under $1600 for the color, dual-disk version. Don't look for it, yet—but when it arrives remember you heard it here, first.

I also saw a brand new Commodore 13-inch monitor that puts out a display you almost can't improve on. They said its patented circuitry was designed specifically for use with Commodore computers, which undoubtedly explains the excellence of its resolution. For $299, which is what I heard it will supposedly sell for, this monitor will be an essential part of your Commodore "component" system. You know, like they sell stereo components?

Not only that, but I saw the first versions of the new "Paperback Book" style software for the VIC 20 and Commodore 64. (Finally, a book that can talk back!) And the first prototypes of the new "touch screen" panel that lets you paint and draw images right on the screen without any special implements. In fact, one of the demonstrators drew some lovely pictures with his fingernail. And, with appropriate software, you can also use the "touch screen" for applications like word processing by, say, touching a menu, then touching the screen. This is almost too futuristic to be believed, I know. But they didn't title this column "unbelievable" for no reason, did they?

Oh, and how about that piano keyboard and Digi-Drum™ for the Commodore 64 they had at the booth? Sure, they were only prototypes, but a three-octave keyboard that really shows off the Commodore 64 SID chip—for that price? And the drum software displays the three drums on a screen, and animates them when they're hit. Nobody else's could do that!

I could just go on and on. For instance, I didn't mention the voice synthesizer cartridge that accepts a variety of smaller cartridges containing a whole range of different voices and vocabularies. I did have to laugh at the demo voice cartridge they were using—a "Valley Girl" voice that said such intelligent things as "No way" and (ugh!) "Gag me with a spoon". Those engineers at Commodore do have a sense of humor, sometimes (strange as it may be). But don't ask when you'll be seeing this voice synthesizer. Nobody will answer—least of all, me. I have to maintain my credibility, you know.

Well, I'm afraid I'll have to sign off, because they told me I have only two-thirds of a page, this time. The indignities one has to suffer in a position like this! And after all I went through to get to the CES at my own expense! So, until next time, keep your ear to the ground. But always make sure your head is attached to it.

C=

Your VIC 20 never had it so good!

VIC 20 owners rejoice! HES presents a complete range of software from our exciting series of games to our professional group of utility and language programs.

Our new cartridge programs include: HES MON, an indispensible monitor for assembly language programmers; HES WRITER, a word processing program; Turtle Graphics, a fun and easy way to learn computer programming; and VIC FORTH, a powerful language that is many times faster than BASIC, yet easier to use than assembly language.

HES is committed to offering high-quality, well-documented computer programs on a continual basis. Look for our cartridge and cassette based software at your local dealer.

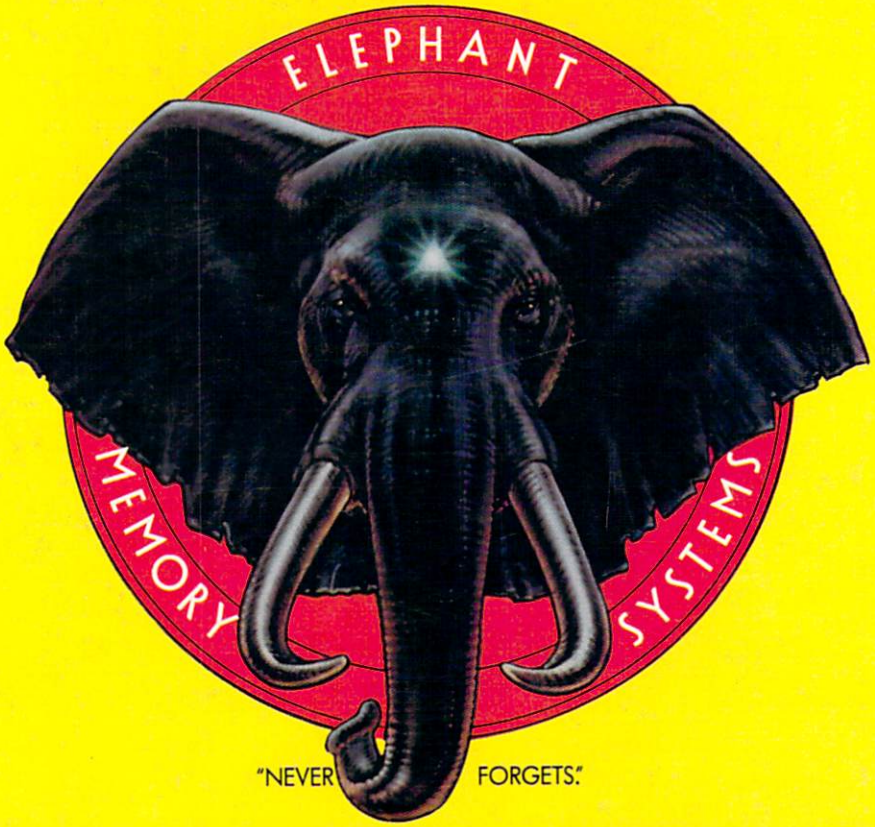
HES is eagerly seeking quality programs for the Commodore product line and outstanding programmers, experienced with these systems.

HES

Human Engineered Software
71 Park Lane
Brisbane, California 94005
Telephone 415-468-4110

VIC 20 is a registered TM of Commodore

REMEMBER:



MORE THAN JUST ANOTHER PRETTY FACE.

Says who? Says ANSI.

Specifically, subcommittee X3B8 of the American National Standards Institute (ANSI) says so. The fact is all Elephant™ floppies meet or exceed the specs required to meet or exceed all their standards.

But just who is "subcommittee X3B8" to issue such pronouncements?

They're a group of people representing a large, well-balanced cross section of disciplines—from academia, government agencies, and the computer industry. People from places like IBM, Hewlett-Packard, 3M, Lawrence Livermore Labs, The U.S. Department of Defense, Honeywell and The Association of Computer Programmers and Analysts. In short, it's a bunch of high-caliber nitpickers whose mission, it seems, in order to make better disks for consumers, is also to make life miserable for everyone in the disk-making business.

How? By gathering together periodically (often, one suspects, under the full moon) to concoct more and more rules to increase the quality of flexible disks. Their most recent rule book runs over 20 single-spaced pages—listing, and insisting upon—hundreds upon hundreds of standards a disk must meet in order to be blessed by ANSI. (And thereby be taken seriously by people who take disks seriously.)

In fact, if you'd like a copy of this formidable document, for free, just let us know and we'll send you one. Because once you know what it takes to make an Elephant for ANSI . . .

We think you'll want us to make some Elephants for you.

ELEPHANT.™ HEAVY DUTY DISKS.

For a free poster-size portrait of our powerful pachyderm, please write us.
Distributed Exclusively by Leading Edge Products, Inc., 225 Turnpike Street, Canton, Massachusetts 02021
Call: toll-free 1-800-343-6833; or in Massachusetts call collect (617) 828-8150. Telex 951-624.